BIBLIOTECA BASICA INFORMATICA

DBASE II Y DBASE III

20



INGELEK

BIBLIOTECA BASICA INFORMATICA

DBASE II Y DBASE III

20

INGELEK

Director editor:
Antonio M. Ferrer Abelló.

Director de producción:
Vicente Robles.

Coordinador y supervisión técnica:
Enrique Monsalve.

Colaboradores:
Angel Segado
Casimiro Zaragoza
Fernando Ruíz
Francisco Ruíz
Jesús Pedraza
Juanjo Alba Ríos
Margarita Caffaratto
María Angeles Gálvez
Marina Caffaratto
Masé González Balandín
Patricia Mordini

Diseño:
Bravo/Lofish.

Dibujos:
José Ochoa.

ISBN del tomo: 84-85831-57-8
ISBN de la obra: 84-85831-31-4
Fotocomposición: Pérez Díaz, S. A.
Imprime: Héroes, S. A.
Depósito Legal: M-4860-1986
Precio en Canarias, Ceuta y Melilla: 380 pts.



INDICE

PROLOGO

El dBASE (II y III hacen referencia a versiones distintas) es una base de datos. Las bases de datos (database) son un conjunto de datos estructurados y organizados de manera que puedan ser fácilmente conservados, puestos al día, modificados, encontrados, elaborados e impresos. Uno de los ejemplos más característicos de las bases de datos es la guía de teléfonos, formada por una serie de informaciones ligadas unas a otras.

No disponiendo de base de datos, cada procedimiento que prevea la memorización y la gestión de los datos debe escribirse en un lenguaje de programación (BASIC, Pascal, Cobol, etc.), y cada sucesivo cambio de los datos almacenados requiere una modificación del programa redactado anteriormente, con la consiguiente pérdida de tiempo y los notables gastos que esto crea. Con la llegada de las bases de datos las operaciones de puesta al día, modificación, elaboración y búsqueda es posible hacerlas en pocos minutos.

Un sistema de manejo de bases de datos (DBMS, Data Base Management System) está constituido por los elementos siguientes:

- programas (escritos en uno de los lenguajes de programación) que permiten realizar la elaboración de la base de datos (creación, puesta al día, modificaciones, búsquedas, etc.);
- lenguaje DDL (Data Definition Language), que permite definir la estructura de cada base de datos;
- lenguaje DML (Data Manipulation Language), que permite

mantener y utilizar cada base de datos anteriormente definida en su estructura.

El dBASE es un DBMS relacional.

En este sistema, los datos se presentan con una estructura de tipo tabla. Cada tabla es memorizada como un archivo en el cual los registros son las líneas horizontales y los campos son las columnas verticales. Ya que las operaciones de manipulación de los datos (insertar, borrar, etc.) se basan, por lo tanto, en un solo operador (la tabla), la utilización de este sistema por parte del usuario es muy sencilla.

La búsqueda de los datos consiste en comparar los campos entre ellos, según criterios estándar y lógicos (selección, proyección, conjunción, etc.), y se traduce en un proceso de construcción de tablas y prospectos.

El elemento principal del dBASE es el archivo, donde las informaciones se guardan en registros, subdivididos en campos, todos del mismo formato.

Después de crear la estructura del archivo se puede realizar la puesta al día con el añadido de nuevos registros. Los registros grabados en el archivo se pueden volver a ordenar con los específicos procesos de ordenación (sort) o de catalogación en base a campos determinados como claves. Los datos insertados y ordenados pueden aparecer en la pantalla tecleando las correspondientes órdenes de búsqueda y selección y se pueden imprimir según el formato deseado.

Con los datos numéricos contenidos en una base de datos también se pueden realizar operaciones contables, matemáticas, estadísticas, etc.

El dBASE realiza las tareas más diversas en empresas o en estudios profesionales, como archivo de datos, para gestionar trabajos de secretaría, como archivo bibliográfico...; además, puede programarse para hacer declaraciones fiscales, para efectuar la facturación, la contabilidad, etc.

El dBASE es un sistema óptimo para el manejo de datos. El dBASE II se adapta muy bien a pequeñas y medianas bases de datos, formadas hasta por 3/4000 registros, con ordenadores personales que tengan por lo menos 256 Kbytes de memoria RAM (para ser eficaz) y con memoria de masa de hasta 2 millones de Kbytes.

El dBASE III representa verdaderamente el futuro; está en condiciones de manejar bases de datos de grandes dimensiones con óptimos resultados en ordenadores personales que tengan, por lo menos, 512 Kbytes de memoria RAM.

CAPITULO I

APROXIMACION AL dBASE

Características del dBASE II y del dBASE III

El dBASE II y el dBASE III están producidos por ASHTON TATE de Culver City (California).

El dBASE II, gestionado por microprocesadores 8080-8085-Z80 y 8088/8086, puede utilizarse en ordenadores de 48 o más Kbytes de RAM, para los primeros, y 128 Kbytes o más para los segundos, con los sistemas operativos CP/M y MS/DOS; éste está disponible tanto para el IBM PC como para todos los ordenadores compatibles con éste, incluido el Olivetti M24.

El dBASE III, gestionado por microprocesadores 8088, puede utilizarse en sistemas de 256 Kbytes de RAM y requiere el sistema MS/DOS 2.0 o una versión superior; actualmente está disponible para el IBM PC y para algunos de los compatibles con él.

El dBASE III está protegido por el sistema Prolok, que obliga a tener el disco del sistema en el drive A cuando se empieza el programa. A diferencia del dBASE II está integrado con un programa de ayuda (HELP) y por un programa interactivo (ASSIST), que funciona como "piloto automático".

Objeto de un sistema de manejo de bases de datos (DBMS)

Un sistema como el dBASE es muy diferente a un sistema de manejo de ficheros. ¿Por qué?

La configuración de un sistema de manejo de ficheros es la expresada en la figura 1, de manera que los ficheros de nóminas

Características	dBASE II	dBASE III
Hardware		
Microprocesador	8080, Z80, 8088/86	8088, 8086
Memoria mínima	48 (128) KBytes	256 Kbytes
Sistema operativo	CP/M 80 y 86	PC-DOS 2.0 (y superiores)
Sistema operativo	MS/DOS	UNIX
Software		
Dimensiones de los campos y de los registros		
Número de campos por registro	32	128
Número de caracteres por registro	1000	4000 (para 512 Kbytes)
Dimensiones del archivo		
Número de registros por archivo	65535	1.000.000.000
Número de archivos abiertos simultáneamente	2	10
Precisión numérica		
Límite máximo de los números naturales	$1.8*10^{63}$	$1*10^{308}$
Límite mínimo de los números naturales	$1*10^{-62}$	$1*10^{-307}$
Exactitud numérica	10	16
Memorias y variables		
Número de variables	64	256
Número de bytes para las variables	1536	6000

son procesados por los programas de nóminas y los programas de contabilidad procesan los ficheros contables. Pero si en un momento dado quisiéramos combinar datos de los dos ficheros, nos veríamos obligados a hacer un nuevo programa y a crear un fichero global.

Un sistema de manejo de bases de datos integra todos los datos, tal y como se muestra en la figura 2.




Figura 1.—Configuración de un sistema de manejo de ficheros.

En este caso es el sistema de manejo de bases de datos (DBMS) el que manipula los datos, no los programas en particular. De esta forma, todas las aplicaciones tienen acceso a todos los datos.

Anteriormente hablábamos de la posibilidad de combinar datos de nóminas y contabilidad (en nuestro ejemplo). Esto, para el caso de un sistema de manejo de ficheros, supondría el hacer un nuevo programa y un nuevo fichero; sin embargo, si contamos con un DBMS, sólo tendremos que escribir un nuevo programa de acceso, pero los datos no tendrán que ser reestructurados.

De la misma forma, si quisiéramos añadir un nuevo dato a un registro, usando un sistema de manejo de ficheros, sería necesario modificar el programa y el fichero de datos. Sin embargo, usando DBMS, cualquier cambio o adición en los datos no implica modificación en los programas que no manejen esos datos en concreto.

Estructura del dBASE

En las figuras 3 y 4 aparecen, respectivamente, un ejemplo elemental de base de datos (ficheros de gestión "manual") y la estructura general de una base de datos cualquiera.

El dBASE (en sus versiones II y III) es un sistema de manejo de bases de datos de tipo relacional, cuya estructura está constituida por archivos compuestos por registros del mismo formato que, a su vez, se dividen en campos.

En un DBMS relacional, los datos están estructurados en forma de tablas:

Nº empleado	Nombre	Cargo	Sueldo	DNI	Estado civil
1 2	Pepe Pérez Lina Glez.	Director Secretaria	3.000.000 1.500.000	52.356 5.632.478	Viudo Soltera

Cada fila (línea) de la tabla es un REGISTRO, cada columna es un CAMPO del registro.

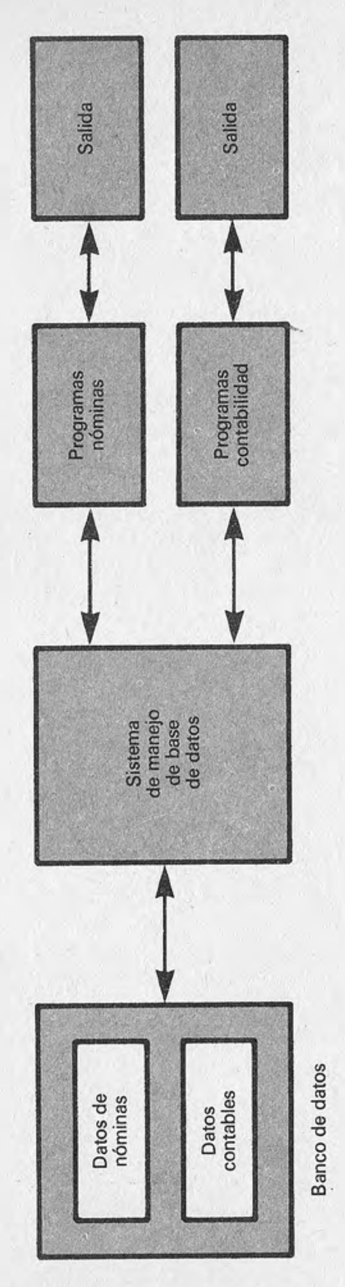


Figura 2.—Estructuración en el caso de un sistema de manejo de bases de datos (DBMS).



Figura 3.—El fichero DIRECCIONES representa un archivo en el que cada ficha es un registro formado por los campos siguientes: nombre, dirección, ciudad, código postal y teléfono.

Todas las entradas en una misma columna son del mismo tipo. Cada fila es única. Cada entrada en la tabla debe ser un valor simple (no se admiten conjuntos, array...).

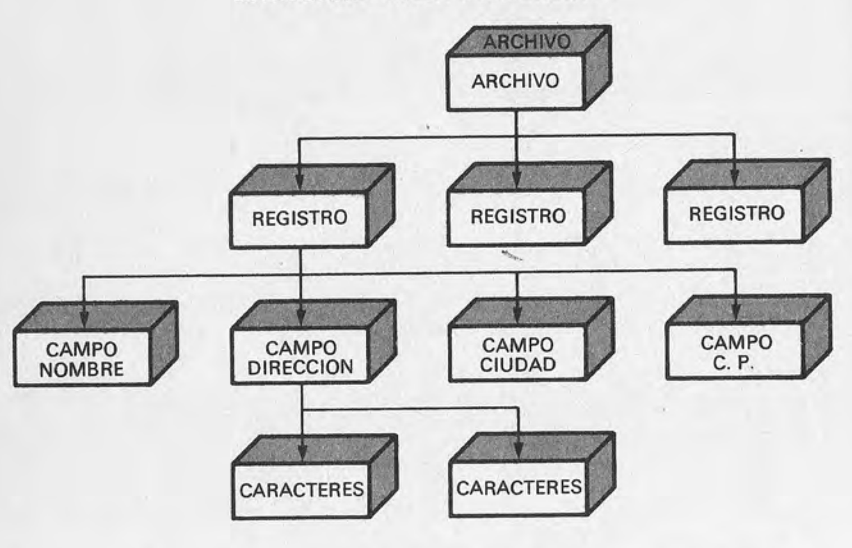
En algunas bases de datos el orden en que se introducen los datos será el que se mantenga para luego recuperarlos. Con el dBASE usted puede organizar los datos usando los comandos **SORT** o **INDEX.**

ARCHIVOS.—El dBASE se compone de dos grupos de archivos: los que contienen las órdenes ejecutables y los que están destinados a recoger los datos del usuario. Estos están especificados por una sigla compuesta por tres letras, llamada extensión (Fig. 5).

Los archivos para la elaboración y programación son:

- KEYWORD FILE: contiene las claves para elaborar el sistema entero;
- COMMAND (o APPLICATION PROGRAMS) FILE (.PRG o

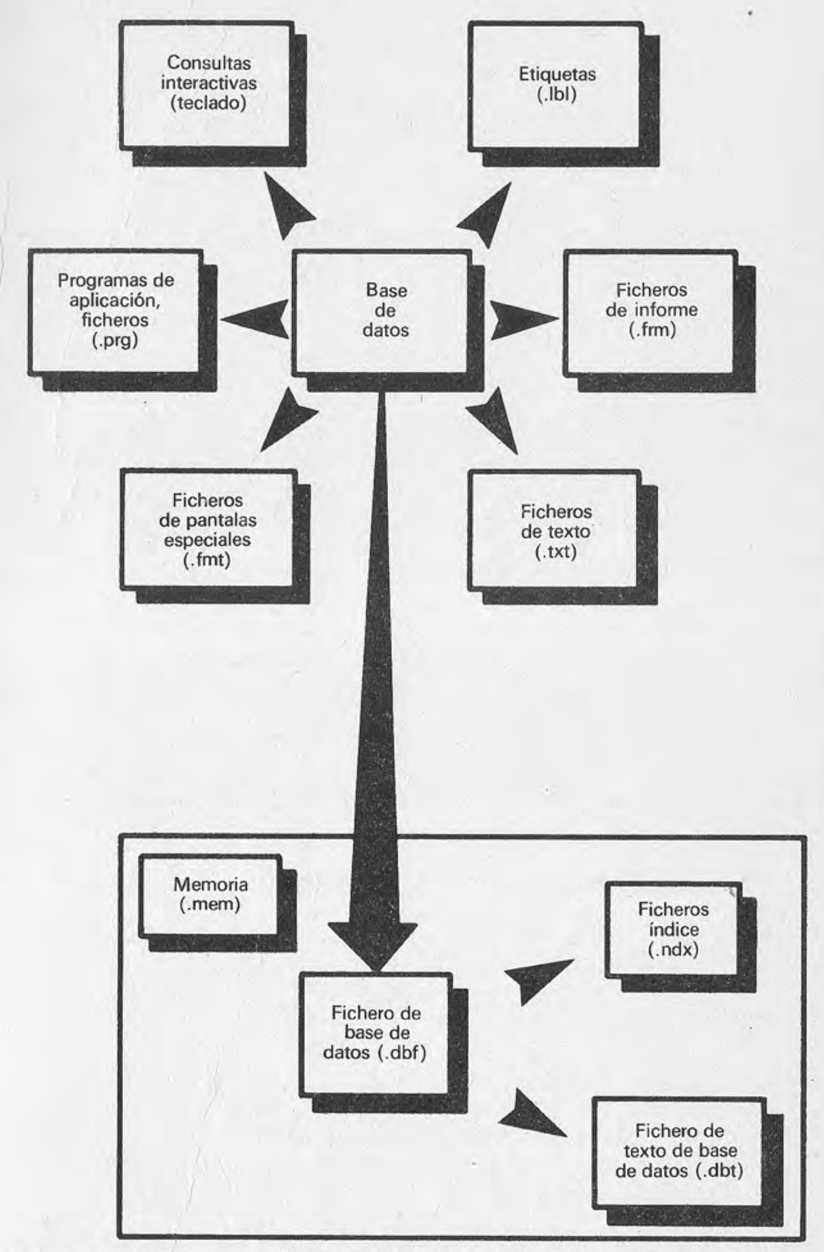
ESTRUCTURA TIPICA DEL ARCHIVO



ARCHIVO			
NOMBRE	DIRECCION	CIUDAD	C. P.
JOSE PEREZ	C/ ORENSE, 4	MADRID	28020
LUIS COLL	C/ DIAGONAL, 121	BARCELONA	08015
PEDRO LOPEZ	C/ ALCALA, 79	MADRID	28006
ANTONIO MILANS	C/ DALI, 52	BARCELONA	08019
ALFREDO MARTIN	C/ GAUDI, 33	BARCELONA	08013

Figura 4.—El archivo está constituido por un conjunto de datos organizados (tabla).
Cada archivo está subdividido en registros (líneas). Cada registro está subdividido en campos (columnas). Cada campo contiene informaciones cuyos caracteres no son divisibles posteriormente.

Figura 5.—En el esquema se presentan los archivos del dBASE. Los archivos de la parte superior, alrededor de "DATABASE", contienen las órdenes para la elaboración de todo el sistema y permiten la programación. KEYBOARD contiene las claves de las tablas. Los REPORT FILES son para preparar relaciones. Los TEXT OUTPUT FILES permiten la edición de textos en caracteres ASCII. CUSTOM SCREEN (FORMAT) FILES sirven para crear plantillas en la pantalla. APPLICATION PROGRAMS (COMMAND) FILES contienen los programas prefijados por el usuario.
En la parte inferior, delimitada por el rectángulo, están representados los archivos de recogida de datos del usuario. MEMORY FILES contiene



las memorias variables. DATABASE FILES sirve para almacenar los datos ordenados según un criterio lógico (alfabético, numérico, etc.). DATABASE TEXT FILE sirve para conservar el contenido de los campos de memoria.

.CMD), sirve para contener una secuencia de sentencias para ejecutar funciones frecuentemente usadas;
- FORMAT (o CUSTOM SCREEN) FILE (.FMT): sirve para formatear datos en la pantalla; sólo puede contener sentencias del tipo: @... SAY;
- TEXT OUTPUT FILE (.TXT): permite la adición de los textos en caracteres ASCII;
- REPORT FORM FILE (.FRM): contiene las instrucciones de los formatos de informes dispuestos por el usuario;
- LABEL FORM FILE (.LBL): sirve para contener las instrucciones de los formatos de tablas dispuestas por el usuario (no está previsto en el dBASE II).

Los archivos para la memorización de los datos son:

- DATA BASE FILE (.DBF): sirve para contener los datos estructurados en registros y campos; su capacidad es asignada por dBASE cuando se crea un nuevo fichero. Cada fichero de este tipo puede almacenar hasta 65.535 registros;
- INDEX FILE (.NDX): sirve para contener los datos de un archivo catalogado; son creados automáticamente por el comando INDEX;
- DATA BASE TEXT FILE (.DBT): sirve para conservar en la memoria el contenido de los campos (no está previsto en dBASE II);
- MEMORY FILE (.MEM): son creados automáticamente cuando se da la orden SAVE de los resultados de la computación, constantes o variables que usted querrá posteriormente. Con SAVE se pueden tomar hasta 64 items de 254 caracteres, luego, mediante un RESTORE, se pueden recuperar estos datos cuando se necesiten.

Los nombres de los archivos no pueden superar los 8 caracteres de longitud y 3 caracteres para la extensión, después del punto. Deben empezar siempre con una letra, y el resto de los caracteres pueden ser cifras, pero no espacios en blanco.

REGISTROS.—Los registros que puede contener cada archivo en el dBASE II son hasta 65.535 y en dBASE III hasta 1 billón (mil millones). El registro del dBASE II se compone de un máximo de 32 campos y puede contener hasta un máximo de 4000 caracteres (en ordenadores de 512 Kbytes en adelante).

CAMPOS.—Cada campo del registro puede contener hasta 254 caracteres (el del dBASE III puede contener hasta 4000). A cada campo se le asigna un nombre, que no puede superar los 10

caracteres, que debe empezar siempre con una letra y que puede contener en su interior cifras pero no espacios.

TIPOS DE DATOS.—Los datos a introducir en los campos pueden ser de los siguientes tipos:

- Datos marcados con C: se refieren a todos los caracteres del alfabeto ASCII incluidas las letras, los números, los símbolos y los espacios;
- Datos marcados con N: se refieren a todos los números positivos y negativos, a los signos y a las cifras decimales necesarias para efectuar cálculos;
- Datos marcados con D: se refieren a las fechas expresadas en la forma estándar de mes/día/año (m/d/y/);
- Datos marcados con M: se refieren a campos con longitud de hasta 4000 caracteres.

CADENAS DE CARACTERES.—Una cadena está constituida por un conjunto de caracteres alfanuméricos encerrados entre comillas. Una subcadena está constituida por una parte de una cadena (también delimitada por comillas) y está contenida en ella.

VARIABLES DE MEMORIA.—Es un espacio de memoria con denominación propia y destinado a acoger el tipo de datos deseados por el usuario.

El lenguaje dBASE

El dBASE (que se basa en un intérprete) puede usarse con órdenes de acción directa o con series de instrucciones diferidas. En este caso presenta todas las características estructurales de un auténtico lenguaje, que se une a los lenguajes de alto nivel tipo Cobol, Pascal, etc., y que ha sido proyectado para elaborar la base de datos que constituye el núcleo central de dBASE.

He aquí una descripción del lenguaje en sus líneas esenciales (alfabeto, vocabulario y sintaxis):

Alfabeto

Está constituido por todos los datos o caracteres sobre los que el ordenador desarrolla los tratamientos deseados y comprende de los tipos siguientes:

- caracteres alfabéticos: son las 26 letras del alfabeto inglés;
- caracteres numéricos: son los dígitos del 0 al 9;
- caracteres de puntuación y otros símbolos especiales.

Vocabulario del dBASE

Está integrado por palabras o expresiones formadas por uno o más caracteres. Estas palabras se pueden dividir en cuatro categorías: constantes, variables, operadores y palabras reservadas (órdenes, funciones y parámetros).

- Las *constantes* son datos insertados en un programa y cuyo valor no se modifica en el curso de su ejecución. Se usan para proporcionar valores conocidos a las variables, para efectuar comparaciones o para especificar mensajes a imprimir; pueden estar formadas por caracteres alfabéticos, numéricos o lógicos. Cuando están constituidas por cadenas de caracteres alfabéticos se deben incluir entre comillas o entre paréntesis cuadrados.
- Las *variables (o identificadores)* son datos que en el curso de la ejecución del programa pueden asumir valores distintos. Pueden estar formadas también por caracteres alfabéticos o numéricos, datos lógicos o por cadenas de caracteres alfanuméricos.
- Los *operadores* están constituidos por símbolos referentes a la ejecución de las operaciones aritméticas (suma, resta, división, multiplicación), de las operaciones de comparación (mayor, menor, igual, etc.), de las operaciones lógicas (.NOT. .AND. .OR) y de las operaciones de cadena.
- Las *palabras reservadas* incluyen todas las órdenes, funciones y parámetros del dBASE que el ordenador ejecuta después de haberlas reconocido. En general, éstas (llamadas también Keyword) se presentan bajo forma de verbos, porque casi siempre indican una acción a cumplir. Se utilizan tal y como están definidas exactamente, porque en otro caso el ordenador no es capaz de reconocerlas.

Las palabras reservadas comprenden:

a) órdenes: constituidas por todas las palabras reservadas o expresiones necesarias para crear y elaborar una base de datos y su contenido;
b) funciones: son órdenes especiales para efectuar automáticamente operaciones de utilidad y de servicio durante la elaboración de la base de datos, operaciones que sería imposible realizar utilizando las expresiones normales;
c) parámetros: son palabras especificadas por el usuario después de algunas órdenes para activar una determinada función del ordenador.

La sintaxis o formato del dBASE

Define cómo las diferentes expresiones (órdenes, funciones, constantes, variables, parámetros) llegan a formar una única instrucción del programa, equivalente a una frase en el lenguaje común.

El conjunto ordenado de distintas frases necesario para hacer ejecutar un dato de trabajo constituye el programa dBASE.

Como en otros lenguajes, las instrucciones del dBASE se pueden reagrupar en cuatro categorías:

- *declarativas:* no forman parte auténtica de la elaboración pero, situadas generalmente al principio del programa, tienen la función de comunicar al ordenador informaciones sobre la utilización de las variables, de las áreas de memoria, de las órdenes, etc;
- *ejecutables de entrada/salida* (INPUT/OUTPUT): se refieren a todas las instrucciones que efectúan una transferencia de datos de entrada (input) o de salida (output) desde o hacia los periféricos.
- *ejecutables de asignación y elaboración* (cálculo): sirven para asignar valores a las variables establecidas y para efectuar, si es necesario, los sucesivos cálculos.
- *de control* (condicional o iterativo): no ejecutan ningún cálculo ni comunicación alguna con el exterior, pero tienen la finalidad de cambiar la sucesión lógica de ejecución de las instrucciones que componen el programa, como, por ejemplo, IF..THEN ENDIF.

Operaciones iniciales

Después de encender el ordenador y haber cargado el sistema operativo MS/DOS o CP/M, se procede a cargar el dBASE en memoria, lo cual puede hacerse de forma manual o automática.

En la forma manual hay que introducir el disco con el dBASE en el drive A (el de la izquierda o el de arriba en el PC portátil), pulsar dBASE y responder al mensaje de la fecha:

ENTER TODAYS AS mm/dd/yy

Si se quiere introducir, por ejemplo, 1 de febrero de 1986, se tecleará 02/01/86 o bien 02-01-86. En respuesta al mensaje de la hora:

ENTER TIME

hay que introducir primero la hora, después los minutos y al final los segundos, separados por ":".

En la forma automática es suficiente con insertar el disco en el drive A y esperar el mensaje de comienzo.

Después del mensaje:

*** dBASE...Ver...Fecha

el programa está preparado para ser utilizado. En la pantalla aparecerá, en la segunda línea, un punto: es el "prompt" (cursor) del dBASE, junto al cual éste acepta las órdenes que le damos. Para las primeras experiencias le sugerimos empezar con algunos cálculos numéricos. En este caso, el diálogo con el ordenador se instaura por medio del signo de interrogación, seguido de las operaciones aritméticas. Así, por ejemplo, si teclea (no olvide acabar pulsando la tecla de Carriage Return):

.? 5+7

En la pantalla, en la línea siguiente, aparecerá el resultado (12).

Para borrar la pantalla se utiliza ERASE, para retornar al "punto" cuando estamos atascados o deseamos parar debemos pulsar la tecla de ESCAPE y para dejar el dBASE basta teclear QUIT.

Pueden continuar ejercitándose planteando ejemplos de suma o de resta. Los operadores utilizados por el ordenador para las operaciones de multiplicación y de división son, como ya es sabido, "*" y "/".

Por ejemplo, planteen la siguiente multiplicación:

.? 5*3

Después de haber apretado la tecla de RETURN en la pantalla aparecerá el resultado 15.

Ahora prueben a ejecutar la siguiente división:

.? 100/3

En el dBASE II (a diferencia del dBASE III) se obtiene como resultado 33, sin la parte decimal. Para obtenerla es suficiente multiplicar por "1.", seguido por tantos ceros cuantos sean los decimales que se quieran obtener en el resultado.

Así, planteando:

.? 100/3*1.0000

se obtiene 33.3333.

Operadores del dBASE

En las expresiones numéricas se ejecutan primero las operaciones entre paréntesis y después las otras, dando preferencias a las multiplicaciones y divisiones frente a las sumas y restas.

A continuación vamos a ver una lista de los operadores y sus tipos:

Operadores aritméticos

Generan resultados aritméticos:

+ suma
- resta
* multiplicación
/ división
( ) paréntesis para establecer grupos y prioridades

Operadores de comparación

Generan resultados lógicos:

< menor que
> mayor que
= igual
<> o # no igual
<= menor o igual que
>= mayor o igual que
$ operador de subcadenas (si A y B son cadenas de caracteres, A$B será verdadero (TRUE) si, y sólo si, la cadena A es igual a la B o está contenida en B).

Operadores lógicos

Generan resultados lógicos:

.OR. O booleano
.AND. Y booleano
.NOT. NO booleano

Operadores con cadenas

El resultado que generan es una cadena.

\+ concatenación de cadenas
\- concatenación de cadenas, eliminando los blancos que pudiera haber entre ambas

Para terminar el trabajo es necesario teclear la orden QUIT seguida de RETURN.

Edición y teclas de control

Una segunda experiencia a efectuar se refiere a la escritura de los textos (editing) en ambiente dBASE, mientras están en función algunas órdenes como APPEND, EDIT, INSERT, MODIFY COMAND, etc.

Cualquier operación de edición presupone la capacidad de maniobrar a placer el cursor en la pantalla lanzando las órdenes adecuadas desde el teclado. Para iniciar el desplazamiento del cursor es suficiente con apretar las siguientes teclas:

- CTRL E (CTRL A) desplaza el cursor una línea arriba;
- CTRL X (CTRL F) desplaza el cursor una línea abajo;
- CTRL S desplaza el cursor un carácter a la izquierda;
- CTRL D desplaza el cursor un carácter a la derecha;
- CTRL C desplaza el cursor 15 líneas hacia abajo;
- CTRL L desplaza el cursor 15 líneas hacia arriba.

La inserción de texto o palabras nuevas en el texto se obtiene tecleando CTRL-V, que hace aparecer el mensaje INSERT en la primera línea de la pantalla. El nuevo texto se inserta a la izquierda del cursor y provoca el desplazamiento hacia la derecha del ya existente.

La función de inserción se desactiva tecleando otra vez CTRL-V o bien RETURN.

Cuando se quieran insertar nuevas líneas es necesario teclear CTRL-N, que va desplazando las líneas hacia abajo mientras se va insertando.

El borrado de una parte del texto se hace con las siguientes teclas:

- CTRL G borra el carácter indicado por el cursor;
- CTRL Y borra el texto a la derecha del cursor;
- CTRL T elimina una línea entera y desplaza hacia arriba las siguientes.

Programas específicos del dBASE III: HELP, ASSIST, CONVERT

A diferencia del dBASE II, el dBASE III está provisto de dos programas muy útiles para el usuario inexperto (Fig. 6).

El primero es HELP, que facilita la utilización de distintas órdenes. Se activa tecleando HELP y la orden de la que se piden explicaciones.

Durante el trabajo de elaboración de la base de datos, el usuario puede interrumpirlo y activar la orden HELP pulsando la tecla F1, que hace aparecer una explicación exhaustiva en la parte superior de la pantalla; apretando F1 de nuevo la pantalla vuelve a su posición normal.

La segunda orden es ASSIST, que funciona como un auténtico piloto automático. Se activa sencillamente tecleando ASSIST y

TECLAS FUNCIONALES DEL DBASE III

TECLA F1	=	HELP (AYUDA)
TECLA F2	=	ASSIST
TECLA F3	=	LIST (LISTADO)
TECLA F4	=	DIR
TECLA F5	=	DISPLAY STRUCTURE (VISUALIZACION ESTRUCTURA)
TECLA F6	=	DISPLAY STATUS (VISUALIZACION ESTADO)
TECLA F7	=	DISPLAY MEMORY (VISUALIZACION MEMORIA)
TECLA F8	=	DISPLAY (VISUALIZACION)
TECLA F9	=	APPEND (ADICION)
TECLA F10	=	EDIT (EDICION)

Figura 6.—Significado de las teclas de la función estándar del dBASE III; las teclas 3 al 10 son definibles por el usuario.

<RE TURN> y durante el trabajo de elaboración de la base de datos se activa y desactiva apretando F2.

Además de las explicaciones de cada orden, está provisto de líneas en negativo donde, a continuación de las elecciones sugeridas por los modelos, los usuarios pueden construirse las órdenes trozo por trozo.

El dBASE III tiene un programa específico para la conversión de los programas predispuestos por el dBASE II. Este programa, denominado CONVERT, requiere una memoria superior a 256 Kbytes (320 o más), dado que su funcionamiento está situado enteramente en la memoria central.

CAPITULO II

ESTRUCTURA DEL dBASE

Creación de la estructura del archivo

Antes de poder almacenar en memoria los datos que nos interesan es necesario crear el archivo adecuado, estructurando los campos que constituyen los registros.

Para crear esta estructura hay que llevar a cabo las siguientes operaciones:

- teclear la orden CREATE;
- cuando aparezca la solicitud "ENTER FILENAME:" escribir el nombre del archivo (y pulsar <RETURN>, como siempre). También puede escribir el nombre al lado de CREATE;
- cuando aparezca la solicitud:
  ENTER RECORD STRUCTURE AS FOLLOWS:
  FIELD NAME,TYPE,WIDTH,DECIMAL PLACES
  001
  hay que definir los campos y, una vez cumplida esta operación, pulsar, <RETURN>, después de lo cual preguntará si deseamos insertar ahora el valor de los registros (INPUT DATA NOW?);
- después de haber respondido que sí y completado los registros, apretando <RETURN> se vuelve al menú principal.

Veamos el proceso más detalladamente:

- CREATE
  visualiza el mensaje siguiente: "ENTER FILENAME:"

Después de este mensaje es necesario escribir el nombre del archivo, que le aconsejamos que sea significativo y corresponda lo máximo posible a los datos que va a contener. No puede tener más de ocho caracteres. Para asignarle el nombre se teclea:

B: Nombre

Si no le ponemos "extensión" (punto y tres caracteres), el dBASE le asignará ".DBF". A continuación aparecerá:

```
ENTER RECORD STRUCTURE AS FOLLOWS:
FIELD  NAME,TYPE,WIDTH,DECIMAL PLACES
001
```

Con este mensaje el sistema solicita la definición de las características de los campos, ateniéndose a las siguientes reglas:

- NAME

  Aquí se introduce el nombre del campo, que puede estar constituido por caracteres (letras/dígitos), siempre que empiece por una letra; no se deben dejar espacios en blanco en medio. El nombre puede tener hasta 10 caracteres.

- TYPE

  En él se define el tipo de datos que se introducirán en el campo. Pueden ser:

  de tipo "C" para datos alfabéticos;
  de tipo "N" para datos numéricos;
  de tipo "L" para datos lógicos;
  de tipo "D" para las fechas (sólo en el dBASE III);
  de tipo "M" para los campos-archivos MEMO (sólo en el dBASE III)

- WIDTH (dimensión)

  Aquí se define la longitud del campo. La dimensión máxima de un campo es de 254 caracteres.

  Cuando se trata de datos numéricos enteros la longitud de este campo será el máximo número de dígitos que se espera que contenga. Para números decimales son requeridos dos tipos de longitud: el primero será el número máximo de dígitos que se espera contenga el campo, incluyendo el punto decimal; el segundo se refiere al número de dígitos que son permitidos a la derecha del punto decimal, y se indica en el campo siguiente.

- DECIMAL PLACES (decimales)

  En él se pueden definir todos los decimales que se deseen después de los enteros. La coma o punto decimal es considerada como un carácter por lo que, por ejemplo, "9999.99" está compuesto por siete caracteres.

¡Atención! Cuando se comete un error durante la estructuración del archivo (por ejemplo, un nombre con excesivos caracteres), aparece el mensaje

BAD NAME FIELD

La entrada deberá repetirse.

No siempre han de completarse todos los datos; así, para un número entero no hará falta indicar los decimales y bastará, por ejemplo, dar: VALOR,N,5.

Las operaciones descritas se repiten hasta que se haya completado la estructura del archivo. Una vez acabada la operación dando a la tecla <RETURN>, aparece el siguiente mensaje:

INPUT DATA NOW?

que, si se contesta con Y, ofrece la posibilidad de tener en seguida la marca "?" de las entradas de datos, mientras que si se contesta con N, devuelve al nivel de órdenes del dBASE (prompt ".").

Ejercicios

Los recomendados especialmente a los que posean el programa dBASE, pero su seguimiento será muy útil e instructivo para todos. La representación se ha hecho conforme aparece en el dBASE II, aunque algunas palabras que en éste aparecen en minúsculas se han puesto en mayúsculas para mejorar la legibilidad, al igual que hemos obviado los "cambios de pantalla" que se producen después de ciertas órdenes.

Creación del archivo VENTAS

Supongamos que queremos crear un archivo que contenga los datos más significativos de las ventas realizadas a los clientes:

El archivo se llamará VENTAS; a cada venta se le reservará un registro que, a su vez, se dividirá en los siguientes campos (indicamos también sus características):

• Campo 1	Nombre del cliente	C	8
• Campo 2	Dirección	C	12
• Campo 3	Ciudad	C	11
• Campo 4	Número de factura	C	3
• Campo 5	Fecha de la factura	C	10
• Campo 6	Precio	N	6
• Campo 7	IVA	N	5
• Campo 8	Total	N	6

He aquí cómo aparece la sesión interactiva en la pantalla:

CREATE - Creación del archivo VENTAS

```
     CREATE    B: VENTAS

ENTER RECORD STRUCTURE AS FOLLOWS:

     FIELD     NAME,TYPE,WIDTH,DECIMAL PLACES
     001       NOMBRE, C,8
     002       CALLE, C,12
     003       CIUDAD, C,11
     004       NRFACT, C,3
     005       FEFACT, C,10
     006       PRECIO, N,6
     007       IVA, N,5
     008       TOTAL, N,6
     009

INPUT DATA NOW? N
```

Para mayor comodidad hemos realizado los mensajes del dBASE respetando las entradas del usuario.

Creación del archivo BIBLIOGRAFIA

Supongamos que queremos crear un archivo que recoja los datos bibliográficos de los volúmenes de nuestra biblioteca.

Al archivo se le asigna el nombre BIBLIO y a cada volumen

se le reserva un registro que, a su vez, se subdivide en los siguientes campos:

• Campo 1	Autor	C	15
• Campo 2	Título del libro	C	40
• Campo 3	Editorial	C	15
• Campo 4	Año de edición	C	5
• Campo 5	Posición en la biblioteca	C	5
• Campo 6	Resumen del tema	C	60

CREATE - Creación del archivo BIBLIO

```
     CREATE    B: BIBLIO

ENTER RECORD STRUCTURE AS FOLLOWS:

     FIELD     NAME,TYPE,WIDTH,DECIMAL PLACES
     001       AUTOR, C,15
     002       TITULO, C,40
     003       EDITORIAL, C,15
     004       ANOEDIC, C,5
     005       POSICION, C,5
     006       RESUMEN, C,60
     007

INPUT DATA NOW? N
```

Creación del archivo DIRECCIONES

A continuación crearemos la estructura del archivo DIRECCIONES formado por los siguientes campos:

• Campo 1	Apellidos	C	25
• Campo 2	Nombre	C	25
• Campo 3	Dirección	C	30
• Campo 4	Cp	C	5
• Campo 5	Ciudad	C	10
• Campo 6	Teléfono	C	10
• Campo 7	Código de identificación fiscal	C	15

```
CREATE - Creación del archivo DIRECCIONES

    CREATE    B: DIRECCIONES

ENTER RECORD STRUCTURES AS FOLLOWS:

    FIELD     NAME,TYPE,WIDTH,DECIMAL PLACES
    001       APELLIDOS, C,25
    002       NOMBRE, C,25
    003       DIRECCION, C,30
    004       CIUDAD, C,6
    005       TELEFONO, C,10
    006       CODFISC, C,15
    007

INPUT DATA NOW? N
```

Creación del archivo AGENDA

Por último proponemos la estructura del archivo AGENDA, compuesta por los siguientes campos:

- Campo 1 Fecha C 10
- Campo 2 Compromisos C 40
- Campo 3 Citas C 40
- Campo 4 Llamadas C 40
- Campo 5 Vencimientos C 40

```
CREATE - Creación del archivo AGENDA

    CREATE    B: AGENDA

ENTER RECORD STRUCTURES AS FOLLOWS:

    FIELD     NAME,TYPE,WIDTH,DECIMAL PLACES
    001       FECHA, C,10
    002       COMPROMISOS, C,40
    003       CITAS, C,40
    004       LLAMADAS, C,40
    005       VENCIMIENTOS
    006

INPUT DATA NOW? N
```

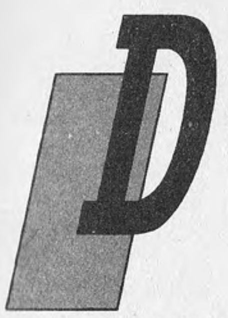

Se puede utilizar también el comando CREATE para la formación de un nuevo fichero a partir de uno ya existente:

CREATE nombre-de-fichero2 FROM nombre-de-fichero1 de esta forma, el fichero2 es creado con la misma estructura del fichero1.

CAPITULO III

DENTRO DEL ARCHIVO

Introducción de los datos: USE



Después de haber estructurado los registros del archivo, se pueden introducir los datos en los diferentes campos establecidos, llevando a cabo las operaciones siguientes:

- teclear la orden USE, escribir el nombre del archivo donde se introducirán los datos y pulsar <RETURN>;
- teclear la orden APPEND;
- aparecerá: RECORD#00001 con la especificación de los campos; se procede entonces a introducir el primer dato. Continuar la inserción de los datos; para concluir pulsar dos veces <RETURN>;
- teclear la orden LIST para comprobar los registros introducidos;
- teclear la orden QUIT para cerrar las operaciones del dBASE.

Veamos detalladamente cada una de estas fases:

- USE nombre-del-archivo

La orden sirve todas las veces que se llama a un archivo con el que operar (aquí USE nos sirve para introducir los datos, dado que el archivo todavía está vacío). Si el archivo ya estuviera abierto, el dBASE respondería con el mensaje

FILE IS CURRENTLY OPEN

• APPEND

La pantalla se borrará y aparece lo siguiente:

RECORD#00001
NOMBRE DEL CAMPO: :
NOMBRE DEL CAMPO: :
NOMBRE DEL CAMPO: :

Obviamente, en lugar de NOMBRE DEL CAMPO se tendrán los nombres asignados en la fase CREATE a cada campo (ejemplo: NOMBRE, CALLE, CIUDAD, etc.; ver también los "ejercicios" al final del capítulo). Los dos puntos (:) finales señalan el límite derecho de cada campo.

La introducción de los datos en el registro debe ser hecha de manera secuencial, campo tras campo.

Si se ha usado el campo en toda su longitud, el cursor se coloca automáticamente en el campo siguiente; si no, para pasar al de abajo basta con apretar <RETURN>.

Cuando se hayan completado todos los campos, el programa pasa automáticamente al registro siguiente.

Para concluir la sesión de entrada basta con apretar <RETURN> cuando el cursor se encuentra en un primer campo vacío.

• LIST

Ahora hay que verificar los datos introducidos.

Tecleando la orden LIST aparecen en la pantalla los datos en el orden del número de los registros que aparece a la izquierda, al principio de cada registro.

• QUIT

Para concluir la sesión (sin perder todos los datos) hay que teclear la orden QUIT.
Aparecerá el mensaje

END RUN dBASE
A>

después de lo cual se puede apagar el ordenador.

Otra aplicación del comando USE es especificar una base de datos para que opere y se la asocie con ficheros indexados.
Ejemplo:

.USE base-de-datos INDEX nombre, ciudad.

Inserción intermedia de nuevos registros : INSERT

Se pueden añadir nuevos registros al final del archivo tecleando APPEND, pero si se quieren introducir nuevos registros entre dos ya existentes se utiliza la orden INSERT, que sólo permite la inserción de un registro cada vez en la base de datos.

El cuadro de las operaciones que hay que ejecutar es el siguiente:

- escribir USE nombre-del-archivo-a-llamar;
- teclear LIST o DISPLAY ALL para comprobar los registros del archivo;
- escribir el salto (GO RECORD o bien GOTO) al registro antes o después del cual tendrá que llevarse a cabo la inserción;
- teclear INSERT BEFORE (para insertar antes) o INSERT (para insertar después);
- cuando aparezca la estructura del registro se procederá a la introducción de los datos;
- pulsar <RETURN> para concluir la inserción.

Veamos algunas nuevas órdenes:

. GO RECORD n (GOTO n)

Con "n" se indica el número del registro que precederá al que se va a insertar.

. INSERT BEFORE visualiza lo siguiente:

RECORD#000n
NOMBRE DEL CAMPO: :
NOMBRE DEL CAMPO: :
NOMBRE DEL CAMPO: :

A partir de este momento se puede proceder a la inserción normal de los datos y, una vez completado el registro, basta con pulsar la tecla <RETURN>. El dBASE registra y numera el nuevo registro.

Con un <RETURN> ulterior se vuelve al nivel de órdenes, evidenciado por la aparición del punto (el "prompt" del dBASE).

Borrado de registros: DELETE

Cuando sea necesario eliminar uno o más registros del archivo de la base de datos, se puede utilizar la orden DELETE.

Con DELETE el registro no se elimina físicamente del disco, sino que es marcado con un asterisco entre el número del registro y el primer campo, para ser eliminado después, definitivamente, con la orden PACK.

El registro así marcado no puede ser copiado, catalogado o clasificado. La operación RECALL puede ser usada para recuperar registros marcados.

Las operaciones que hay que hacer son:

- escribir USE nombre-del-archivo-a-llamar;
- teclear DELETE RECORD y el número del registro que hay que borrar, o bien DELETE FOR y la expresión que determinará el borrado o no del registro y apretar <RETURN>. Con esta orden se marca el registro o registros que hay que borrar;
- bien teclear RECALL RECORD y el número del registro marcado que hay que llamar, o bien RECALL FOR y la expresión elegida y pulsar <RETURN>; así se recuperan los registros señalados antes para su borrado;
- o bien teclear PACK para eliminar los registros marcados con la orden DELETE.

Concretemos algunas instrucciones:

- DELETE <órdenes específicas>

Este comando puede completarse con algunas órdenes específicas, según las necesidades. Son las siguientes:

ALL:	Todos los registros son marcados con el carácter *;
RECORD n:	Sólo el registro especificado con el número "n" es marcado con el asterisco.
NEXT n:	Todos los registros después del número especificado son marcados con el asterisco.

- DELETE FOR <condiciones>

Con esta orden se pueden formular las siguientes condiciones, que permiten el borrado selectivo de los registros:

= igual
> mayor que...
< menor que...
>= mayor o igual que...
<= menor o igual que...
<> distinto

¡Cuidado! Todas las constantes alfanuméricas en juego deben delimitarse entre dos comillas " ".

Veamos algunos ejemplos de la utilización de la orden DELETE FOR:

a) DELETE FOR NOMBRE="Alfa A"
borra todos los registros que contienen por campo NOMBRE la cadena "Alfa A".
b) DELETE FOR IMPORT>100000
borra todos los registros que contengan un campo IMPORT mayor que 100000.

La orden DELETE puede unirse a las siguientes funciones lógicas, que pueden utilizarse también durante la programación normal:

.NOT. negación lógica
.AND. conjunción lógica
.OR. elección lógica entre dos operaciones posibles

Además, los paréntesis indican qué operaciones o expresiones deben efectuarse con anterioridad.

- RECALL <condición> y RECALL FOR <expresión>

Con estas órdenes se anulan todos los borrados en curso y las marcas son eliminadas.

- PACK

Confirma el borrado de los registros marcados y los elimina definitivamente del disco, sin posibilidad alguna de recuperación. Por lo tanto, tengan cuidado y ejecuten PACK sólo cuando estén seguros de no haber cometido errores (con esta finalidad, una comprobación con LIST o LIST FOR nunca viene mal).

Si el fichero al que se le aplica el comando PACK es indexado, el propio comando se encarga de ajustar los índices cuando se borra algún registro.

Corrección de datos de los registros: EDIT

La orden dispuesta para esta actividad es EDIT. ¿En qué consiste materialmente la modificación de los datos en el interior de los registros? Se trata de la superposición de nuevos caracteres alfanuméricos, introducidos por el teclado, a los ya existentes.

EDIT puede usarse junto a una serie de teclas, función que nos permite movernos, de forma más práctica, en el interior del registro.

Las operaciones que hay que hacer son:

- escribir USE nombre-del-archivo-a-llamar;
- teclear LIST o bien DISPLAY ALL para visualizar los registros y localizar el número del que se va a corregir;
- escribir EDIT n (número del registro a corregir) para visualizar el registro que hay que corregir. Modificarlo;
- teclear CTRL-W para guardar en memoria los cambios efectuados.

- EDIT n

Este comando, con la definición de "n" (número del registro), sirve para reclamar el registro que contiene los datos que hay que modificar:

RECORD # 000n
NOMBRE DEL CAMPO: :
NOMBRE DEL CAMPO: :
NOMBRE DEL CAMPO: :

Si no se especifica "n", entonces dBASE preguntará por las coordenadas del registro que se quiere editar.

Una vez aparecidos los datos se puede efectuar la corrección con la ayuda de las siguientes teclas de función:

CTRL-C escribe los cambios en el disco y llama al registro siguiente para su edición;
CTRL-R escribe los cambios en el disco y llama al registro anterior para su edición;
CTRL-Q cierra las operaciones sin hacer cambios y vuelve al nivel de órdenes normales;
CTRL-V inserta caracteres y espacios entre una palabra y otra y entre un carácter y otro;
CTRL-U marca los registros que se desea cambiar, es decir, opera como la orden DELETE y al pulsarla de nuevo actúa al revés.

- CTRL-W

Esta orden (que forma parte de EDIT) permite almacenar en memoria todas las modificaciones efectuadas en los registros.

¡Cuidado! En algunos teclados la tecla CTRL corresponde a ALT.

Concluidas todas las operaciones, hay que ejecutar la orden QUIT para cerrar el archivo.

Modificar datos de los campos: CHANGE

Cuando surge la necesidad de completar o modificar los datos contenidos en uno o más campos de los registros de un archivo, se utiliza la orden CHANGE.

Las operaciones que hay que hacer son las siguientes:

- escribir USE nombre-del-archivo-a-llamar;
- teclear LIST o bien DISPLAY ALL para visualizar los registros;
- escribir CHANGE ALL FIELD <nombre del campo> FOR <condición> para visualizar el campo a modificar;
- aparecerá el contenido antiguo del campo, que puede modificarse pulsando antes <RETURN>;
- introducir el nuevo contenido y dar <RETURN> para cerrar la operación de modificación.

Examinemos las órdenes nuevas.

- CHANGE ALL FIELD Nombre-del-campo FOR <condición>

Con esta orden se define el campo que se quiere modificar y, si se desea, también se pueden añadir condiciones para modificar sólo registros que respondan a preguntas preestablecidas.

Además, es posible llamar simultáneamente a distintos campos. Estos deben de estar separados con una coma cuando se formule la orden. Por ejemplo: supongamos que queremos completar todos los campos que estén vacíos. La orden podría ser la siguiente:

CHANGE ALL FIELD Apellido FOR Apellido="　　"

Así serán visualizados y propuestos sólo los registros que contengan el campo Apellido vacío.

RECORD # 000n
Apellido: :
CHANGE?

(Véase el ejemplo al final del capítulo).

Después de haber echado un vistazo al antiguo contenido del campo es suficiente con pulsar <RETURN> para pasar a la fase siguiente, en la que el dBASE hace aparecer la palabra:

TO

al lado de la cual se puede insertar ahora el nuevo dato que sustituye al viejo, confirmándolo nosotros con <RETURN>:

NOMBRE DEL CAMPO: Texto nuevo:
CHANGE?

Con un <RETURN> posterior se cierra la operación y se memorizan las modificaciones en el registro.

Modificar los datos de todos los registros: REPLACE

A veces puede ser útil modificar algunos datos de todos los registros de un archivo. Lo bueno es que no se tiene que tratar sólo de sustituciones de constantes, sino que pueden ser cálculos que implican a los campos. Por lo tanto, la operación es de sustitución y valoración.

Así, si en un archivo se han introducido las facturas de venta sin el IVA y, a continuación, se quisiera calcular el total aumentado por el IVA, se podría utilizar la orden REPLACE para esta opeación. Pero ¡cuidado!, ya que sólo se modificarán los campos que hayan sido introducidos con la orden APPEND. Los que se han incorporado con INSERT no son tomados en consideración.

Si no se especifica el rango donde se aplica este comando, entonces REPLACE actúa solamente en el registro actual.

Las operaciones a cumplir son las siguientes:

- escribir USE nombre-del-archivo-a-llamar;
- teclear LIST o DISPLAY ALL para visualizar los registros;
- escribir REPLACE ALL <nombre del campo> WITH <nuevo dato> para sustituir los viejos datos en el campo especificado;
- teclear LIST o DISPLAY ALL para comprobar el archivo ya modificado.

- REPLACE ALL <nombre del campo> WITH <nuevos datos>

Esta orden permite sustituir los datos del campo especificado por los nuevos datos.

REPLACE puede aplicarse a las claves de ficheros indexados: después de haber utilizado el comando se borrará la clave antigua y la nueva se introducirá en su mismo lugar.

También se pueden efectuar sustituciones totales del archivo especificando algunas condiciones; de esta manera, las correcciones y los cambios tienen lugar solamente en los campos que corresponden a las condiciones especificadas:

- REPLACE ALL <nombre del campo> WITH <nuevos datos> FOR <condición>

Ejemplo: Si se tiene un importe con IVA incluido y se desea quitarlo basta con dar la orden REPLACE de la siguiente forma:

REPLACE ALL Importe WITH Importe * 100/112

La confirmación de que se ha efectuado la orden se tiene con el siguiente mensaje:

000n REPLACEMENT(S)

donde n (precedido por ceros no significativos) es el número de sustituciones efectuadas por nuestro dBASE.

- *Ejercicios*

Haciendo referencia al archivo VENTAS procedemos a poner ejemplos de la utilización de las órdenes APPEND, INSERT, DELETE, EDIT, CHANGE y REPLACE, examinadas en el capítulo.

APPEND - Introducción de registros en el archivo VENTAS

Introducimos los cinco registros siguientes en el archivo VENTAS.

- USE B: VENTAS
- APPEND

RECORD	# 00001
NOMBRE	: Alfa A. :
CALLE	: Calle Madrid, 5 :
CIUDAD	: Barcelona :
NRFACT	: 111 :

APPEND - Introducción de registros en el archivo VENTAS

```
DFACT          : 1985/05/06 :
PRECIO         : 100000 :
IVA            :      0 :
TOTAL          :        :

RECORD         # 00002

NOMBRE         : Beta B.   :
CALLE          : Calle Avila, 5 :
CIUDAD         : Sevilla    :
NRFACT         : 333 :
DFACT          : 1985/05/10 :
PRECIO         : 200000 :
IVA            :      0:
TOTAL          :       0:

RECORD         # 00003

NOMBRE         : Gamma C. :
CALLE          : Calle Orense, 5 :
CIUDAD         : Madrid     :
NRFACT         : 222 :
DFACT          : 1985/08/05 :
PRECIO         : 300000 :
IVA            :      0:
TOTAL          :       0:

RECORD         # 00004

NOMBRE         : Delta D.   :
CALLE          : Huesca, 5 :
CIUDAD         : Madrid     :
NRFACT         : 444 :
DFACT          : 1985/07/85 :
PRECIO         : 400000 :
IVA            :      0:
TOTAL          :       0:

RECORD         # 00005

NOMBRE         : Alfa A.  :
CALLE          : Calle Madrid, 5 :
CIUDAD         : Barcelona :
```


APPEND - Introducción de registros en el archivo VENTAS

```
NRFACT         : 166 :
DFACT          : 1985/09/25 :
PRECIO         : 500000 :
IVA            :      0:
TOTAL          :       0:
```

INSERT - Introducción intermedia de un registro en el archivo VENTAS

Supongamos que entre los registros 4 y 5 añadimos una venta efectuada al cliente Gamma C.

```
.USE B:VENTAS
.GO RECORD 4
.INSERT

RECORD         # 00005

NOMBRE         : Gamma C :
CALLE          : Calle Madrid, 5 :
CIUDAD         : Barcelona :
NRFACT         : 161 :
DFACT          : 1985/05/25 :
PRECIO         : 600000 :
IVA            :      0:
TOTAL          :       0:
```

DELETE - Borrado de registros en el archivo VENTAS

Continuando el ejemplo anterior, supongamos que borramos el registro 5 insertado anteriormente y referente al cliente Gamma C.

```
.USE B:VENTAS
.DELETE RECORD 5
00001 RECORD(S) DELETED
.PACK
```

DELETE - Borrado de registros en el archivo VENTAS

```
00005 RECORD COPIED
.LIST
```

00001	Alfa A.	Calle Madrid, 5	Barcelona	111	1985/05/06	100000	0	0
00002	Beta B.	Calle Avila, 5	Sevilla	333	1985/05/10	200000	0	0
00003	Gamma C.	Calle Orense, 5	Madrid	222	1985/08/05	300000	0	0
00004	Delta D.	Calle Huesca, 5	Madrid	444	1985/07/20	400000	0	0
00005	Alfa A.	Calle Madrid, 5	Barcelona	166	1985/09/25	500000	0	0

EDIT - Corrección de registros en el archivo VENTAS

Supongamos que queremos cambiar la fecha de la factura del cliente Alfa A, cambiando 1985/05/06 por 1985/06/05.

Esta modificación se obtiene maniobrando con los caracteres de comprobación descritos en el capítulo.

```
.USE B:VENTAS
.EDIT 1

RECORD          # 00001

NOMBRE          : Alfa A.  :
CALLE           : Calle Madrid, 5 :
CIUDAD          : Barcelona :
NRFACT          : 111 :
DFACT           : 1985/06/05 :
PRECIO          : 100000 :
IVA             :        0:
TOTAL           :

CTRL-W
```

CHANGE - Modificación de datos de los campos en el archivo VENTAS

Supongamos que queremos cambiar el nombre de la calle del cliente Alfa.

```
.USE B:VENTAS
.CHANGE FIELD CALLE
```


CHANGE - Modificación de datos de los campos en el archivo VENTAS

```
RECORD # 00001

CHANGE? Calle Madrid, 5
TO Calle Alcalá, 5
CHANGE? ↵
```

REPLACE - Corrección de datos de los registros en el archivo VENTAS

Supongamos que queremos insertar el importe del IVA y el importe TOTAL en el archivo VENTAS.

```
.USE B:VENTAS
.REPLACE ALL IVA WITH PRECIO*0,12
00005 REPLACEMENT(S)
.LIST
```

00001	Alfa A.	Calle Madrid, 5	Barcelona	111	1985/05/06	100000	12000	0
00002	Beta B.	Calle Avila, 5	Sevilla	333	1985/05/10	200000	24000	0
00003	Gamma C.	Calle Orense, 5	Madrid	222	1985/08/05	300000	36000	0
00004	Delta D.	Calle Huesca, 5	Madrid	444	1985/07/20	400000	48000	0
00005	Alfa A.	Calle Madrid, 5	Barcelona	166	1985/09/25	500000	60000	0

```
.REPLACE ALL TOTAL WITH PRECIO+IVA
00005 REPLACEMENT(S)
.LIST
```

00001	Alfa A.	Calle Madrid, 5	Barcelona	111	1985/05/06	100000	12000	112000
00002	Beta B.	Calle Avila, 5	Sevilla	333	1985/05/10	200000	24000	224000
00003	Gamma C.	Calle Orense, 5	Madrid	222	1985/08/05	300000	36000	336000
00004	Delta D.	Calle Huesca, 5	Madrid	444	1985/07/15	400000	48000	448000
00005	Alfa A.	Calle Madrid, 5	Barcelona	166	1985/09/25	500000	60000	560000

Resumen de las sintaxis completas

Vamos a resumir ahora la sintaxis de los comandos vistos, expresándola en su forma más global, incluyendo las diversas opciones, señaladas entre paréntesis cuadrados. Tanto éstos como los signos "<" y ">" no deberán incluirse al teclear la orden.

USE

USE [<fichero de base de datos>]

```
USE <fichero de base de datos> INDEX <fichero indexado>
    [, <fichero indexado>,... <fichero indexado>]
```

INSERT

```
INSERT [BEFORE] [BLANK]
```

DELETE

```
DELETE [<extensión>] [FOR <expresión>]
                     [WHILE <expresión>]
DELETE FILE <nombre de fichero>
```

RECALL

```
RECALL [<extensión>] [FOR <expresión>]
                     [WHILE <expresión>]
```

PACK

```
PACK
```

EDIT

```
EDIT [n]
```

CHANGE

```
CHANGE [<extensión>] FIELD <lista> [FOR <expresión>]
```

REPLACE

```
REPLACE [<extensión>] <campo> WITH <expresión>
        [, <campo> WITH <expresión>]
            [FOR <expresión>] [NOUPDATE]
            [WHILE <expresión>]
```


CAPITULO IV

PONIENDO ORDEN EN LOS ARCHIVOS

Ordenación de los registros de un archivo: SORT

Cuando en un archivo se desea una nueva secuencia, acompañada de una renumeración total de los registros, es necesaria la orden SORT.

Su sintaxis es:

```
SORT ON <campo> TO <fichero>
[ASCENDING o DESCENDING]
```

Genera un nuevo archivo (el archivo destino) en el que los registros provenientes del archivo original son ordenados físicamente según el valor de un campo particular elegido como clave de la ordenación.

El orden de los datos puede ser ascendente o descendente, según los valores literales y numéricos deban ser considerados en orden creciente (de la A a la Z y del 0 al 9) o decreciente (de la Z a la A y del 9 al 0). Cada secuencia diferente requiere una ordenación nueva. Si no se especifica nada, asume por defecto el orden ascendente.

SORT utiliza para la ordenación el código ASCII; así, la cadena "JUAN" es más pequeña que "Juan":

JUAN < Juan es verdad

SORT es una operación que emplea mucho tiempo en el dBASE II, mientras que en el dBASE III se efectúa en pocos minutos.

Las operaciones necesarias son las siguientes:

- escribir USE nombre-del-archivo-original-a-ordenar;

- teclear LIST o bien DISPLAY ALL para visualizar los registros;
- escribir SORT ON <nombre del campo> TO <nombre del archivo ordenado> para ordenar el archivo original mediante su transferencia a un nuevo archivo;
- escribir USE <nombre del archivo ordenado>;
- teclear LIST o bien DISPLAY ALL para visualizar los registros ordenados.

Describamos la orden nueva:

- SORT ON nombre_del_campo TO nombre_del_archivo_ordenado

Esta orden define el nombre del campo clave según el cual se ordenarán los datos y el nombre del archivo en el que se memorizarán los datos ordenados.

Cuando se haya completado la operación aparecerá el siguiente mensaje:

SORT COMPLETE

La sesión podrá continuar con la orden USE (para llamar y abrir el archivo ordenado) seguido de LIST o bien DISPLAY ALL (para comprobar que el archivo nuevo es correcto).

Si se trabaja con archivos grandes es conveniente comprobar antes si en el disco hay suficiente espacio, dado que con la orden SORT se ocupa el mismo espacio que con el original.

Cuando se dispone de poco espacio puede venir bien borrar el archivo original con la orden DELETE (después de haber ordenado SORT obviamente).

Indexación de los registros de un archivo: INDEX

La indexación de los datos es una variante útil de la ordenación, de la que se diferencia, sin embargo, sustancialmente. En efecto, crea un archivo nuevo (especificado como NDX) que contiene sólo índices de registro, ordenados según el campo clave, pero no en el orden del número de registro.

La sintaxis es:

INDEX ON <nombre del campo> TO <nombre del fichero>

Su efecto es crear el fichero <nombre del fichero> que contiene punteros a los registros del fichero que se desea catalogar.

La orden SORT cumple dos operaciones: una es la de ordenar alfabética o numéricamente el archivo y la otra la de renumerar totalmente los registros en base a la nueva posición asumida.

La orden INDEX, en cambio, desarrolla sólo el trabajo de la ordenación alfabética o numérica. Pero si bien no tiene la ventaja de la renumeración, sí tiene la de poder ser utilizada para la búsqueda veloz mediante la orden FIND, que funciona sólo con registros indexados (Fig. 1).

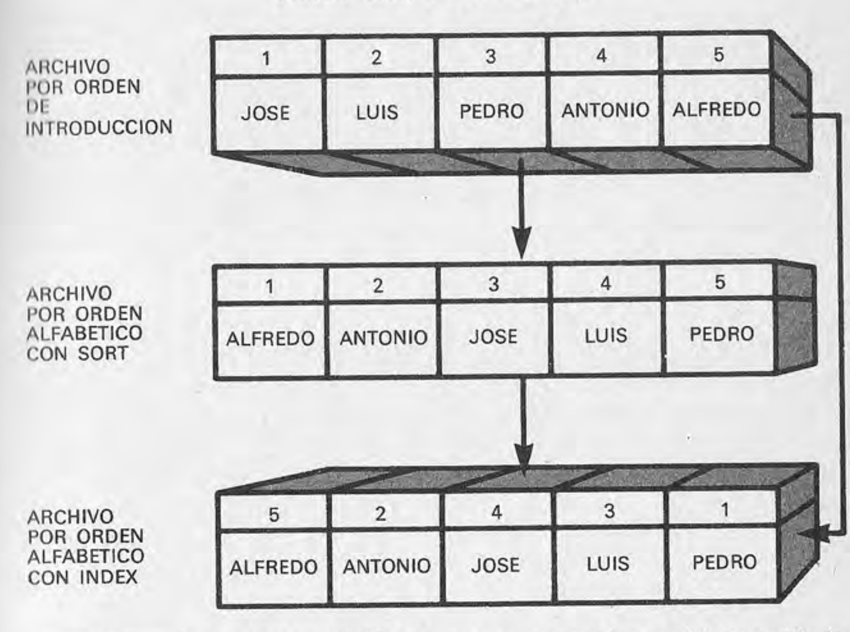


Figura 1.—La orden SORT ordena y renumera los registros según la nueva posición asumida; la orden INDEX desarrolla la clasificación según el criterio elegido (alfabético), pero sin renumerar los registros.

Cada archivo creado con la orden INDEX asume como sufijo o extensión la partícula NDX, mientras que los creados con la orden SORT siguen siendo de tipo DBF. En efecto, los archivos ".NDX" contienen sólo punteros al registro del archivo original (que permanece invariable). De esta manera la operación INDEX requiere un espacio muy limitado.

Para indexar un archivo se llevan a cabo las siguientes operaciones:

- escribir USE Nombre-del-archivo-a-catalogar;
- teclear LIST o bien DISPLAY ALL para visualizar los registros;
- escribir INDEX ON <Nombre del campo> TO <Nombre del nuevo archivo catalogado> para llamar al archivo catalogado;
- escribir USE <Nombre del archivo fuente> INDEX <Nombre del archivo catalogado> para llamar al archivo catalogado;
- teclear LIST o bien DISPLAY ALL para visualizar los registros catalogados.

\- INDEX ON <Nombre del campo> TO <Nombre del nuevo archivo>

Con esta orden se define el campo según el cual se ordenará el archivo.

Si se quieren ordenar según varios campos simultáneamente (claves de ordenamiento múltiple) bastará con indicar los campos elegidos unidos por el signo (+) de la siguiente forma:

INDEX ON <nombre del campo>+
<nombre del campo> TO INDEX

Pero cuidado: si uno de los campos es de tipo numérico (Type N) hay que transformarlo antes en campo de tipo alfabético, anteponiendo al nombre del campo la función STR:

INDEX ON <nombre del campo>+
STR (nombre del campo, Número) TO <nombre del archivo>

Por ejemplo:

STR (Importe, 10,2)

Una vez completada la operación se visualiza el siguiente mensaje:

000n RECORD INDEXED

Escribimos ahora:

USE <Nombre del archivo> INDEX <Nombre del archivo catalogado>

Esta orden sirve para llamar y abrir el archivo catalogado. Finalmente, con el habitual

LIST o bien DISPLAY ALL

se puede comprobar el archivo catalogado listándolo en la pantalla.

En el dBASE III, con la orden SET INDEX TO se pueden abrir y manipular hasta siete archivos INDEX simultáneamente.

Ejercicios

Refiriéndonos al archivo VENTAS, probemos ahora a ordenarlo con la orden SORT y después a catalogarlo con la orden INDEX.

SORT - Ordenación del archivo VENTAS

Procedemos a la ordenación del archivo ventas según el campo NOMBRE.

```
.USE B:VENTAS
.SORT ON NOMBRE TO B:SNOMBRE
SORT COMPLETE
.USE B:SNOMBRE
.LIST
```

00001	Alfa A.	Calle Madrid, 5	Barcelona	111	1985/06/05	100000	12000	112000
00002	Alfa A.	Calle Madrid, 5	Barcelona	166	1985/09/25	500000	60000	560000
00003	Beta B.	Calle Avila, 5	Sevilla	333	1985/05/10	200000	24000	224000
00004	Delta D.	Calle Huesca, 5	Madrid	444	1985/07/20	400000	48000	448000
00005	Gamma C.	Calle Orense, 5	Madrid	222	1985/09/85	300000	36000	336000

INDEX - Catalogación del archivo VENTAS

Usaremos el campo NRFACT

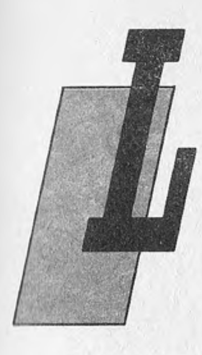
```
.USE B:VENTAS
.INDEX ON NRFACT TO B:CTNRFACT
00005 RECORDS INDEXED
.USE B:VENTAS INDEX B:CTNRFACT
LIST
```

00001	Alfa A.	Calle Madrid, 5	Barcelona	111	1985/06/05	100000	12000	112000
00005	Alfa A.	Calle Madrid, 5	Barcelona	166	1985/09/25	500000	60000	560000
00003	Gamma C.	Calle Orense, 5	Madrid	222	1985/05/08	300000	36000	336000
00002	Beta B.	Calle Avila, 5	Sevilla	333	1985/05/10	200000	24000	224000
00004	Delta D.	Calle Huesca, 5	Madrid	444	1985/07/20	400000	48000	448000

CAPITULO V

BUSCANDO DATOS Y REGISTROS

Búsqueda de registros mediante su número



La búsqueda de los registros de un archivo puede efectuarse según diferentes criterios. Lo más fácil es, precisamente, buscarlo mediante su número, desplazando el puntero por el interior del archivo.

Para efectuar esta búsqueda las operaciones que se pueden ejecutar, después de haber activado los habituales USE, seguidos eventualmente de LIST o DISPLAY ALL, son:

- GO TOP para saltar al registro inicial del archivo;
- GO BOTTOM para saltar al registro final del archivo;
- GOTO n para saltar al registro "n" del archivo;
- SKIP -/+ n para saltar respecto al registro en el que nos encontremos, "n" registros: hacia atrás (-) o hacia delante (+);
- LIST NEXT n o bien DISPLAY NEXT n para visualizar los registros siguientes al registro "n".

A éstas seguirá DISPLAY + para visualizar el registro encontrado.

Estas órdenes sirven para desplazar al puntero dentro del archivo a la búsqueda de registro.

```
GO TOP
DISPLAY
```

El puntero salta al registro inicial del archivo y con DISPLAY este registro se visualiza.

Para movernos al último registro:

GO BOTTOM
DISPLAY

Se puede ir a un registro específico usando esta expresión:

GOTO n
DISPLAY

Igualmente se podría acceder a ese registro tecleando un número únicamente y eliminando GOTO.

SKIP +/n
DISPLAY

El puntero salta los "n" registros sucesivos si se utiliza el signo + y retrocede "n" registros si se utiliza el signo –, a partir de la posición en que nos encontremos.

DISPLAY ALL

Este comando muestra el archivo e interrumpe la lista cada 15 registros, haciendo aparecer el mensaje "WAITING". Para que continúe la lista es suficiente apretar cualquier tecla.

LIST NEXT n o DISPLAY NEXT n

Para visualizar directamente los datos que se encuentran en el registro "n" especificado, se utilizan estos comandos.

Búsqueda de los registros a través de los datos

La búsqueda directa por número de registro es incómoda cuando, como pasa a menudo, no se conoce su posición o bien se quieren ver los registros que posean ciertos datos.

Este tipo de búsqueda se efectúa con las órdenes LIST FOR, DISPLAY FOR y LOCATE FOR a través de la expresión que existe en el interior del registro.

Fundamentalmente las tres órdenes tienen la misma función pero se diferencian en la forma en la que presentan los datos encontrados en la pantalla. En efecto, mientras LIST FOR y DISPLAY FOR muestran los datos en la secuencia de lista, LOCATE FOR visualiza sólo el número del primer registro que contiene los

datos buscados; la visualización del registro queda a elección del usuario.

Las operaciones a seguir, una vez abierto el archivo con USE, son:

LIST FOR <expresión> o DISPLAY FOR <expresión>

para visualizar la lista de los registros que contienen la expresión especificada;

LOCATE FOR <expresión>

para visualizar el primer registro que contiene la expresión. Después se podrá teclear CONTINUE para visualizar los registros siguientes que contienen la misma expresión.

Con las órdenes:

LIST FOR <condición> o DISPLAY FOR <condición>

se efectúa la búsqueda de los registros a través de la indicación de las condiciones. En cuanto a éstas, en primer lugar tienen que corresponder exactamente al contenido del campo. Además, deben ajustarse a las funciones de comparación que enumeramos a continuación:

= igual
> mayor que ...
< menor que ...
>= mayor o igual que ...
<= menor o igual que ...
<> distinto que ...

Con ellas se pueden obtener órdenes como las siguientes:

LIST FOR NOMBRE >= "Alfa"

aparecerán listados todos los registros que contengan en NOMBRE Alfa.

DISPLAY FOR CIUDAD="Barcelona"

aparecerán listados todos los registros que contengan en CIUDAD Barcelona.

¡Cuidado! Repetimos que todos los caracteres alfanuméricos contenidos en la condición tienen que aparecer entre comillas " " (si el campo es numérico no sirven).

LIST FOR IMPORTE>100000

aparecerán listados todos los registros que contengan un importe mayor que 100000.

DISPLAY FOR IMPORTE<200000

aparecerán listados todos los registros que contengan un importe menor que 200000.

LIST FOR NOMBRE<>"Gamma"

aparecerán listados todos los registros que NO contengan el nombre Gamma.

DISPLAY FOR PRECIO<> 400000

aparecerán listados todos los registros que contengan importes distintos de 400000.

Además, es posible componer órdenes complejas combinándolas con las funciones lógicas y adoptando eventualmente los paréntesis para fijar las operaciones que deben ejecutarse primero:

.NOT. la condición no es tomada en consideración;
.AND. las dos condiciones deben de ser satisfechas;
.OR. basta con que una de las dos condiciones sea satisfecha.

Se pueden usar las siguientes órdenes:

LIST FOR CIUDAD="Barcelona".AND.PRECIO=100000

hace el listado de los registros que contienen en CIUDAD Barcelona y en PRECIO 100000.

LOCATE FOR NOMBRE="Beta B".OR. NRFACT=222

hace el listado de los registros que contienen en NOMBRE Beta B o en número de la Factura 222.

Búsqueda a través de cadenas y subcadenas

Para entender cómo funciona esta búsqueda hay que recordar los conceptos de cadena y subcadena. La cadena es parte del contenido de un campo. En efecto, puede estar formada por uno

o más caracteres, una o más cifras, o una o más palabras puestas entre comillas (" "). La subcadena es una parte de la cadena.

Por ejemplo, el nombre "López" es una cadena, mientras que "ope" es una subcadena suya (recuerden las muñecas rusas, compuestas por una muñeca que contiene otras). Con el dBASE es posible efectuar la búsqueda de los datos utilizando las cadenas y las subcadenas con las órdenes LIST, DISPLAY y LOCATE. Las operaciones son como las que explicamos a continuación y, obsérvese, se caracterizan por el símbolo $ antepuesto al nombre del campo:

- USE <Nombre del archivo a llamar> y una de las tres siguientes:
- LIST FOR "expresión" $ < nombre del campo> para hacer la lista de todos los registros que contengan la expresión establecida;
- DISPLAY FOR "expresión" $ <nombre del campo> para hacer la lista de todos los registros que contengan la expresión establecida;
- LOCATE FOR "expresión" $ <nombre del campo> para visualizar el número del registro donde se encuentra la expresión indicada;

Veamos algunos ejemplos:

LIST "XX" $ <nombre del campo>

Con esta orden se hace la lista de todos los registros que contengan la cadena especificada por las comillas " ".

LIST FOR "Delta" $NOMBRE

Hace la lista de todos los registros que contengan en nombre DELTA.

DISPLAY FOR "expresión" $ <nombre del campo>
DISPLAY FOR "Ba" $CIUDAD

Hace la lista de todos los registros que contengan la subcadena "Ba" en el campo CIUDAD.

LOCATE FOR "expresión" $ <nombre del campo>
LOCATE FOR "/07" $FECHA

Se sitúa en el registro que contiene la fecha "/07".

DISPLAY

Visualiza el registro encontrado con LOCATE.

Búsqueda de los datos en un archivo indexado

La búsqueda de los registros en el interior de un archivo se puede realizar con la orden FIND, que permite localizar los datos rápidamente en pocos segundos, casi la mitad de los utilizados con las órdenes LOCATE, DISPLAY FOR o LIST FOR. La orden FIND sólo trabaja con archivos indexados y visualiza el número del registro donde se encuentran los datos solicitados.

No siempre es aconsejable utilizar FIND, sobre todo cuando hay en el disco más archivos para gestionar.

Las operaciones a seguir son las siguientes:

- escribir USE nombre-del-archivo-originario INDEX <nombre del archivo catalogado>;
- escribir FIND <número del campo o expresión> para localizar el registro que contiene el campo o expresión especificada;
- teclear DISPLAY para visualizar el registro.

Estas órdenes se detallan a continuación:

— USE <Nombre del archivo> INDEX <Nombre del archivo catalogado>

Sirve para llamar y abrir el archivo catalogado.

— FIND "XXX"

Naturalmente, esta orden tiene que tener en el lugar de "XXX" la especificación justa. El dBASE busca la especificación correspondiente entre los distintos registros y para el puntero en el primer registro que la contiene, visualizando el punto del prompt a la espera de la orden DISPLAY.

— DISPLAY o DISPLAY NEXT n

Visualiza en la pantalla el registro localizado con FIND.

Ejercicios

Refiriéndonos al archivo VENTAS procederemos a ver la utilización de las órdenes GO, GOTO, SKIP, FIND, LIST FOR, DISPLAY FOR y LOCATE FOR, sin subcadena y con ella.

GO y GOTO - Búsqueda de registros en el interior del archivo VENTAS

```
.USE B:VENTAS
.GO RECORD 2
.DISPLAY

00002  Alfa A.   Calle Madrid, 5  Barcelona  111  1985/06/05  100000  12000  112000

.USE B:VENTAS
.GOTO RECORD 3

00003  Gamma C.  Calle Orense, 5  Madrid     222  1985/05/10  200000  24000  224000

.USE B:VENTAS
.GO RECORD 1
.SKIP +3
RECORD: 00004
.SKIP -2
RECORD: 00002
```

FIND - Búsqueda de datos en el interior del archivo VENTAS indexado

Buscamos en el archivo VENTAS indexado los registros que contiene el 2 y el 111 en el número de la factura.

```
.USE B:VENTAS INDEX B:INNFACT
.FIND "2"
.DISPLAY

00003  Gamma C.  Calle Orense, 5  Madrid     222  1985/08/05  300000  36000  336000

.USE B:VENTAS INDEX B:INNFACT
```

FIND - Búsqueda de datos en el interior del archivo VENTAS indexado

```
.FIND "111"
.DISPLAY
```

LIST DISPLAY y LOCATE FOR - Búsqueda de datos en el interior del archivo ventas NO indexado

Ahora buscamos, respectivamente, los registros que contiene el nombre del cliente Alfa, la ciudad de Madrid y los que cumplen varias condiciones en el importe de las facturas.

```
.USE B:VENTAS
.LIST FOR NOMBRE="Alfa A."
```

00001	Alfa A.	Calle Madrid, 5	Barcelona	111	1985/06/05	100000	12000	112000

```
.USE B:VENTAS
.LOCATE FOR CIUDAD="Madrid"
RECORD: 00003
.DISPLAY
```

00003	Gamma C.	Calle Orense, 5	Madrid	222	1985/08/05	300000	36000	36000

```
.CONTINUE
RECORD: 00004
```

00004	Delta D.	Calle Huesca, 5	Madrid	4444	1985/07/20	400000	48000	448000

```
.USE B:VENTAS
.DISPLAY FOR TOTAL<200000 .AND. IVA<20000
```

00001	Alfa A.	Calle Madrid, 5	Barcelona	111	1985/06/05	100000	12000	112000

Y un ejemplo de búsqueda utilizando subcadena.

```
.USE B:VENTAS
.LIST FOR "ta" $NOMBRE
```

00002	Beta B.	Calle Avila, 5	Sevilla	333	1985/05/10	200000	24000	224000
00004	Delta D.	Calle Huesca, 5	Madrid	444	1985/07/20	400000	48000	448000

CAPITULO VI

MANIPULACIÓN DEL ARCHIVO

Copia del archivo

Durante la elaboración normal de un archivo puede ser necesario copiarlo total o parcialmente en otro. Esto es posible hacerlo utilizando la orden COPY TO. Las operaciones a cumplir, después de la apertura del archivo, son:

- Escribir COPY TO <nombre del nuevo archivo> para copiar el archivo;
- Escribir USE <nombre del archivo> para llamar el archivo copiado;
- Teclear LIST STRUCTURE para efectuar el control de la estructura;
- Teclear LIST o DISPLAY ALL para visualizar los registros copiados.

Cuando se ejecuta un COPY sobre un fichero existente, se borran los antiguos datos de este fichero, ya que los nuevos datos son escritos sobre los anteriores.

Veamos detalladamente las operaciones:

COPY TO <nombre del nuevo archivo>

La orden COPY TO sirve para copiar un archivo en otro. Obviamente, el nombre del nuevo archivo debe de ser distinto del original.

El archivo del que se copia puede estar en formato dBASE o en el formato del Sistema, con lo cual habría que especificar "SDF" (System Data File).

Si se especifica la cláusula "SDF", el nuevo fichero estará en formato ASCII estándar.

USE <nombre del nuevo archivo>

Con esta orden se abre el archivo copiado, que es manejado como un archivo normal del dBASE.

LIST STRUCTURE

Esta orden pone de manifiesto la estructura de un archivo, es decir, la subdivisión en campos y su naturaleza. En este caso nos sirve para comprobar que el nuevo archivo tiene, efectivamente, la misma estructura del original.

Además, la orden COPY sirve para copiar los datos en un archivo de tipo TXT. El dBASE está dotado de un sistema de elaboración de textos que permite almacenar en memoria textos extraídos de archivos de tipo ".DBF". La única condición que hay que respetar es que el campo tiene que acabarse con un punto.

Otras opciones de este comando son:

COPY TO <nombre de fichero> STRUCTURE

Con este comando no se hace más que la copia de la estructura.

Una modificación de la anterior es:

COPY TO <nombre de fichero> STRUCTURE FIELD <lista de nombres de campos>

en cuyo caso sólo se copia una parte de la estructura, con los campos que aparezcan en la "lista de nombres de campos".

Modificación de la estructura

A veces puede suceder que surja la necesidad de modificar la estructura, bien añadiendo o quitando campos, bien aumentando o disminuyendo sus dimensiones.

Para evitar la pérdida de datos ya introducidos hay que efectuar una serie de operaciones que enumeramos a continuación:

- escribir USE <nombre del archivo a llamar>;
- teclear LIST STRUCTURE para visualizar la estructura del archivo;
- escribir COPY TO <archivo de servicio> para copiar el archivo original en uno auxiliar;

- escribir USE <nombre del archivo original>;
- teclear MODIFY STRUCTURE para modificar la estructura del archivo originario;
- teclear "Y" cuando aparezca:

  MODIFY ERASES ALL RECORDS... PROCEED? (Y/N)

  para entrar en la estructura a modificar;
- cuando nos presente la estructura de los campos, realizar las modificaciones necesarias;
- teclear CTRL-W para almacenar las modificaciones que hemos llevado a cabo;
- teclear LIST STRUCTURE para visualizar la estructura del archivo modificado;
- escribir APPEND FROM <nombre del archivo de servicio> para transferir los datos desde el archivo de servicio al archivo modificado;
- teclear LIST o bien DISPLAY ALL para visualizar el archivo original modificado, completo de datos;
- escribir DELETE FILE <nombre del archivo de servicio> para borrar el archivo de servicio.

Después de la habitual apertura del archivo con USE, las otras operaciones tienen la función ilustrada a continuación:

— LIST STRUCTURE o bien DISPLAY STRUCTURE

Se visualiza lo siguiente:

```
STRUCTURE FOR FILE : Nombre del archivo. DBF
NUMBER OF RECORDS: 0000n
DATE OF LAST UPDATE: 00/00/00
PRIMARY USE DATABASE
FLD  NAME  TYPE  WIDTH  DEC
001
002
 ...
 ...
**Total**
```

(Obviamente, debajo de NAME, TYPE, etc., aparecerán el nombre, tipo, etc., de los diferentes campos que componen el registro.)

— COPY TO FILE <nombre del archivo de servicio>

Mediante esta orden, el archivo original a modificar es copiado íntegramente en el archivo de servicio.

— USE <nombre del archivo original>

Con esta orden se reabre el archivo original para poder efectuar las modificaciones.

— MODIFY STRUCTURE

Con esta orden se visualiza la siguiente advertencia:

MODIFY ERASES ALL DATA RECORDS... PROCEED? (Y/N)

En efecto, al cambiar la estructura se borran los datos del archivo cuya estructura se modifica. Hay que contestar "Y" sólo cuando se tiene la certeza de haber hecho la copia del archivo de servicio, tal y como dijimos arriba.

Respondemos "Y".

Esta respuesta visualiza en la pantalla lo siguiente:

	NAME	TYPE	LEN	DEC
FIELD 01:				:
FIELD 02:				:
FIELD 03:				:
etc.				

Donde en el interior aparecen todos los campos de la estructura, con su nombre, tipo, dimensiones y decimales correspondientes.

Desde este momento se puede proceder a las modificaciones con la ayuda de las siguientes teclas-función de edición:

- CTRL-N inserta líneas vacías delante de aquella en la que se encuentra el cursor;
- CTRL-T borra los caracteres a la derecha de la posición en que se encuentra el cursor;
- CTRL-Y borra la línea entera y la prepara para insertar la modificación;
- CTRL-R desplaza el cursor una línea hacia adelante;
- CTRL-C desplaza el cursor una página;
- CTRL-Q abandona el proceso sin aceptar los cambios efectuados en el archivo;
- CTRL-W registra los cambios efectuados en el disco.

Por lo tanto, al terminar tecleamos:

CTRL-W

orden que almacena todas las modificaciones realizadas.

Proseguimos con

LIST STRUCTURE o bien DISPLAY STRUCTURE

orden que comprueba que la modificación se ha efectuado en la forma deseada.

Ahora viene la fase de recuperación de los datos:

APPEND FROM <nombre del archivo de servicio>

transfiere los datos del archivo de servicio al archivo original, que ahora posee la nueva estructura. Si en el curso de la modificación se ha borrado un campo, los datos contenidos en él no serán transferidos y, por lo tanto, quedará vacío.

Una vez completada la orden se visualizará el siguiente mensaje:

00n RECORD ADDED

LIST o bien DISPLAY ALL

Lista los datos del archivo modificado.

DELETE FILE <Nombre del archivo de servicio>

Cuando se tenga la certeza de que todo está en su sitio se puede proceder a borrar el archivo de servicio. Una vez completada la orden se visualiza el siguiente mensaje:

FILE HAS BEEN DELETED

En el dBASE III no hace falta hacer la copia del archivo antes de ordenar MODIFY STRUCTURE, porque esta versión crea automáticamente la copia de back-up.

Transferencia de campos de un archivo a otro

Siempre mediante la orden COPY, algunos campos de un archivo pueden transferirse a uno nuevo (cuidado con esto, pues al hacer el COPY la estructura del "nuevo" archivo quedará modifi-

cada, y si no es nuevo perderá toda su anterior información). Para efectuar esta transferencia deben estar separados en la orden por una coma.

Las operaciones a cumplir, después de una eventual apertura con USE y LIST STRUCTURE del original, son las siguientes:

- escribir COPY TO <nombre del nuevo archivo> STRUCTURE FIELD <nombre de los campos a copiar>;
- escribir USE <nombre del nuevo archivo>;
- teclear LIST STRUCTURE para verificar la estructura nueva;
- escribir APPEND FROM <nombre del archivo original> para trasladar datos desde el archivo original al nuevo;
- teclear LIST para verificar el contenido del archivo.

Lo que se obtendrá tras cada paso es:

COPY TO <Nombre del nuevo archivo> STRUCTURE FIELD <Nombre del campo, ..., nombre del campo>

Con esta orden se procede a copiar sólo la estructura de los campos del archivo indicado especificados en la lista formada por los distintos nombres de campo, separados por comas. Además, se copian todas las especificaciones del campo y los decimales.

Con la orden:

USE <nombre del archivo nuevo>

se llama de nuevo al archivo, que queda abierto a las distintas operaciones posibles. Lo lógico será efectuar primero la siguiente:

LIST STRUCTURE o DISPLAY STRUCTURE

con el fin de visualizar la estructura recién creada.

A continuación es indispensable (¡no lo olviden!) hacer lo siguiente:

APPEND FROM <Nombre del archivo>

a través de lo cual se trasladan los datos desde el archivo original al archivo parcial y modificado.

Por último, después de los habituales (opcionales)

LIST o DISPLAY ALL

para concluir las operaciones teclearemos la orden QUIT.

Cambio del nombre y cancelación del archivo

El manejo de un archivo también prevé el cambio de su nombre en cualquier momento. Para hacer esto, el dBASE dispone de la orden RENAME, con la sintaxis siguiente:

RENAME <nombre antiguo del archivo> TO <nombre nuevo del archivo>

Si no se especifica extensión (punto y tres dígitos a continuación del nombre de archivo), dBASE asigna el tipo .DBF.

La eliminación de un archivo del disco se realiza, como ya hemos visto, con la orden

DELETE FILE <nombre del archivo>

Al final de la operación aparece el mensaje de confirmación:

FILE HAS BEEN DELETED

Ni el cambio de nombre ni el borrado son posibles sobre el fichero actualmente abierto ("en USE").

Ejercicios

Volviendo a utilizar el archivo VENTAS, explicamos a continuación la utilización de las órdenes COPY, MODIFY STRUCTURE, COPY TO, STRUCTURE FIELDS, RENAME, DELETE.

COPY - Duplicación de algunos campos del archivo VENTAS en un archivo nuevo

Partiendo de un archivo VENTAS procedemos a crear uno nuevo llamado PRUEBA.

```
.USE B:VENTAS
.COPY TO B:PRUEBA
00005 RECORDS COPIED
.USE B:PRUEBA
.LIST
```

00001	Alfa A.	Calle Madrid, 5	Barcelona	111	1985/06/05	100000	12000	112000
00002	Beta B.	Calle Avila, 5	Sevilla	333	1985/05/10	200000	24000	224000
00003	Gamma C.	Calle Orense, 5	Madrid	222	1985/07/20	300000	36000	336000
00004	Delta D.	Calle Huesca, 5	Madrid	444	1985/07/20	400000	48000	448000
00005	Alfa A.	Calle Madrid, 5	Barcelona	166	1985/09/25	500000	60000	560000

MODIFY STRUCTURE - Modificación de la estructura del archivo VENTAS

Procedemos a modificar el número de caracteres del primer campo desde 8 hasta 10.

```
.USE B:VENTAS
.MODIFY STRUCTURE
MODIFY ERASES ALL DATA RECORDS...PROCEED? (Y/N) Y

          NAME        TYPE    LEN     DEC
FIELD 01: NOMBRE       C      008     000:
FIELD 02: CALLE        C      012     000:
FIELD 03: CIUDAD       C      011     000:
FIELD 04: NRFACT       C      003     000:
FIELD 05: FECFACT      C      010     000:
FIELD 06: PRECIO       N      006     000:
FIELD 07: IVA          N      005     000:
FIELD 08: TOTAL        N      006     000:
FIELD 09: NOTAS        C      010     000:
FIELD 10:                                :
FIELD 11:                                :
FIELD 12:                                :
FIELD 13:                                :
FIELD 14:                                :
FIELD 15:                                :
FIELD 16:                                :
```

(En este punto se variará con el formato de edición lo que se quiera...)

CTRL-W (para cerrar la sesión de modificación de la estructura)

```
APPEND FROM B:PRUEBA
00005 RECORDS ADDED
.USE B:VENTAS
.LIST STRUCTURE
STRUCTURE FOR FILE: B:VENTAS.DBF
NUMBER OF RECORDS: 00005
DATE OF LAST UPDATE: 01/01/86
PRIMARY USE DATABASE

FLD   NAME        TYPE    WIDTH    DEC
001   NOMBRE       C       010
002   CALLE        C       012
```


MODIFY STRUCTURE - Modificación de la estructura del archivo VENTAS

```
003   CIUDAD       C       011
004   NRFACT       C       003
005   FEFACT       C       010
006   PRECIO       N       006
007   IVA          N       005
008   TOTAL        N       006
009   NOTAS        C       010
**TOTAL**                 00074

DELETE FILE B:PRUEBA
FILE HAS BEEN DELETED
```

COPY TO ... STRUCTURE FIELDS - Copia de la estructura del archivo VENTAS en un archivo nuevo

Partiendo del archivo VENTAS procedemos a crear uno nuevo llamado PRUEBA y estructurado con los campos siguientes: NOMBRE, PRECIO, IVA, TOTAL.

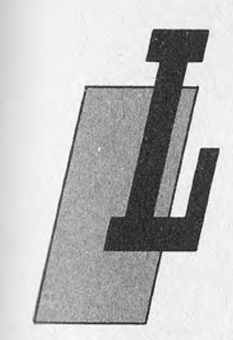
```
.USE B:VENTAS
.COPY TO B:PRUEBA STRUCTURE FIELDS NOMBRE, PRECIO,
  IVA, TOTAL
.USE B:PRUEBA
.LIST STRUCTURE
STRUCTURE FOR FILE: B:PRUEBA DBF
NUMBER OF RECORDS: 00000
DATE OF LAST UPDATE: 01/01/86
PRIMARY USE DATABASE

FLD   NAME        TYPE    WIDTH    DEC
001   NOMBRE       C       008
002   PRECIO       N       006
003   IVA          N       005
004   TOTAL        N       006
**TOTAL**                 00026
```

CAPITULO VII

SELECCION, PUESTA AL DIA Y UNION

Selección de los datos de un archivo



La selección permite la subdivisión de los datos contenidos en un archivo grande (multivolumen) en distintos archivos más pequeños (volumen) y puede resultar muy ventajosa cuando se quieren estructurar sub-archivos para una consulta más rápida.

La creación de estos sub-archivos se hace con la orden COPY TO...FOR, que extrae de un archivo original todos los registros que responden a unas condiciones preestablecidas y los copia en un archivo nuevo. Hay que dar la orden "USE", indispensable si el archivo en el que se va a trabajar no está abierto ya.

La siguiente sintaxis es:

COPY TO <nombre del archivo nuevo de destino> FOR <nombre del campo> <criterio de selección>...<nombre del campo> <criterio de selección>

Esta orden permite extraer los datos de un archivo y transferirlos a otro nuevo, según los siguientes criterios de selección:

= igual que...
> mayor que...
< menor que...
<> distinto de...

De forma análoga a los ejemplos ya vistos hacemos:

USE <nombre del archivo nuevo>

para llamar al nuevo archivo resultante de esta operación de selección, y:

LIST o DISPLAY ALL

para visualizar y comprobar que se han transferido los datos deseados.

Puesta al día de un archivo con los datos de otro

Para actualizar un archivo sin tener que llamar a cada registro, se puede utilizar la orden UPDATE, que modifica el archivo en USE transfiriendo los datos desde un segundo archivo. El archivo en el que se efectuará al puesta al día debe ser catalogado o clasificado según una clave de puesta al día prefijada, que debe escribirse a continuación de UPDATE ON.

El dBASE hace una comparación entre los dos archivos y cuando la clave de puesta al día se corresponde en los dos archivos, los datos del archivo a actualizar son añadidos o sustituidos.

Las operaciones que hay que realizar son las siguientes:

- escribir USE <nombre del segundo archivo que contiene los datos de puesta al día>;
- escribir USE <nombre del archivo a poner al día> INDEX <nombre del archivo catalogado>;
- teclear UPDATE ON <nombre del campo clave> FROM <nombre del segundo archivo> ADD <nombre del campo a añadir> REPLACE <nombre del campo cuyos datos se sustituirán>;
- escribir LIST para listar el nuevo archivo puesto al día.

A continuación se explican estas operaciones:

USE <nombre del segundo archivo>

Esta orden se utiliza para abrir el archivo en el que sustituirán o añadirán datos nuevos.

USE <nombre del archivo a poner al día> INDEX <nombre del archivo catalogado>

Con esta orden se abre el archivo catalogado, en el que tendrá lugar la puesta al día.

UPDATE ON <nombre del campo clave> FROM <nombre del segundo archivo, con los datos de la puesta al día> ADD <nombre del campo a añadir> REPLACE <nombre del campo a sustituir>

Esta es la sintaxis de la orden UPDATE, la cual permite añadir o sustituir los datos en el archivo a actualizar, cogiéndolos del de puesta al día.

Unión de dos archivos

La orden JOIN es una de las más interesantes del dBASE. Su cometido es el de unir dos archivos en un tercero (Fig. 1). El primer archivo es seleccionado como PRIMARY y el segundo como SECONDARY, mediante la utilización de la orden SELECT, que permite la elaboración simultánea de dos archivos en el dBASE II y de diez archivos en el dBASE III.

Una vez establecido cuál es el archivo primario y cuál el secundario se teclea JOIN, que examina cada registro del archivo secundario para comprobar si la expresión es verdadera en relación con el primer registro del archivo primario. En caso afirmativo, el registro se añade a un tercer archivo recién formado.

Después de esto, el puntero del archivo primario se desplaza al segundo registro, que es completamente examinado, y así sucesivamente hasta el final. La unión tiene lugar a través de las siguientes operaciones:

- escribir USE <nombre del archivo primario>;
- escribir SELECT SECONDARY para designar el archivo secundario;
- escribir USE <nombre del archivo secundario>;
- teclear SELECT PRIMARY para abrir el archivo primario;
- escribir JOIN TO <nombre del tercer archivo> FOR <expresión> FIELDS <lista de los campos> para unir los dos archivos;
- escribir USE <nombre del nuevo archivo> para abrir el nuevo tercer archivo;
- LIST para listar el contenido del tercer archivo.

Veamos, paso por paso, las diferentes fases, de las que también habrá ejercicios al final del capítulo.

USE <nombre del archivo primario>

Esta orden sirve para llamar y abrir el archivo que, por defecto, será seleccionado como PRIMARIO.

UNION DE DOS ARCHIVOS

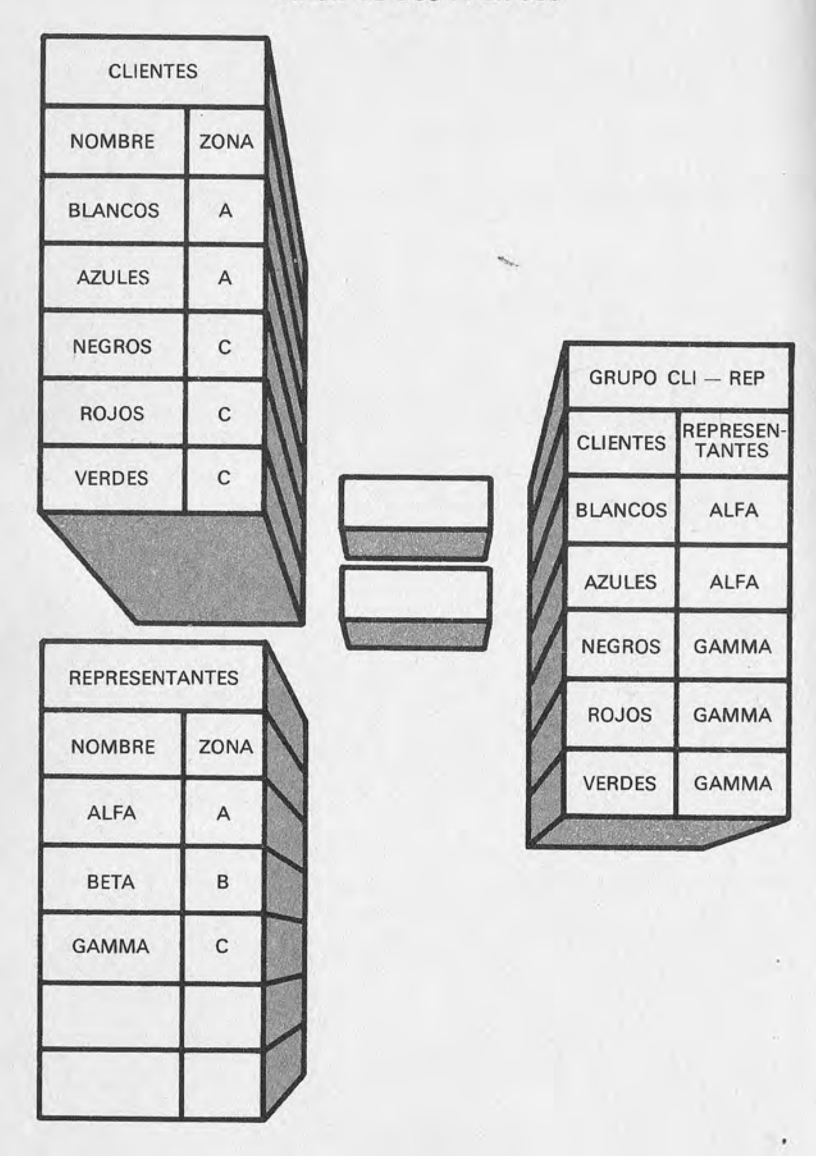


Figura 1.—La orden JOIN permite, por ejemplo, unir el archivo CLIENTES con el archivo REPRESENTANTES para formar el nuevo archivo GRUPO CLI-REP, en función de la condición ZONA.

SELECT SECONDARY

Esta orden sirve para seleccionar el archivo secundario (se especifica con el próximo USE).

USE <nombre del archivo secundario>

Abre el segundo archivo, que se convierte así en SECUNDARIO. De ahora en adelante hay abiertos dos archivos, a los cuales nos podemos referir sencillamente con SELECT PRIMARY o SELECT SECONDARY.

SELECT PRIMARY

Con esta orden se llama al archivo designado como PRIMARIO.

JOIN TO <nombre del tercer archivo> FOR <expresión> FIELD <lista de campos>

Esta es la forma general de la orden JOIN.
Se hace notar (ver ejercicios específicos al final del capítulo) que habiendo dos archivos abiertos ahora se podría tener alguna confusión en el caso de campos homónimos (en la expresión que sigue a FOR). Para evitar equívocos se escribirán los prefijos "P." o "S."; por ejemplo: "S. NOMBRE" designa el campo NOMBRE del secundario, "P. NOMBRE" o también sólo "NOMBRE", el campo NOMBRE del primario.
Si la operación de unión se hace con grandes archivos habrá que esperar un poco. En el dBASE II hay que tener presente las dimensiones de los dos archivos que se unen, porque el resultante no debe superar los 65.535 registros.
Este problema no aparece con el dBASE III, porque el nuevo archivo puede contener hasta un billón de registros (es necesario tener el adecuado espacio físico disponible en el disco, naturalmente).

USE <nombre del tercer archivo>

Esta orden sirve para abrir el tercer archivo así creado.

LIST

Sirve para listar el contenido del tercer archivo.
El nuevo fichero tiene la extensión ".DBF". La estructura de los campos que se copian tiene la misma que los archivos primario y secundario.

En el dBASE III la unión y la elaboración simultánea de dos archivos puede efectuarse también con la orden SET RELATION TO.

Ejercicios

Refiriéndonos siempre al archivo VENTAS, supongamos que queremos hacer las siguientes operaciones de selección, puesta al día y unión.

COPY FOR - Selección de algunos campos del archivo VENTAS

Procedemos a seleccionar y transferir a un archivo llamado SELALFA todas las ventas efectuadas al Sr. Alfa.

```
.USE B:VENTAS
.COPY TO B:SELALFA FOR NOMBRE='Alfa A.'
00002 RECORDS COPIED
.USE B:SELALFA
.LIST
```

00001	Alfa A.	Calle Madrid, 5	Barcelona	111	1985/06/05	100000	12000	112000
00002	Alfa A.	Calle Madrid, 5	Barcelona	166	1985/09/25	500000	60000	560000

UPDATE - Poner al día el archivo VENTAS

Después de haber creado un archivo denominado VENT2 y haber insertado dos registros, procedemos a transferirlos al archivo VENTAS existente.

```
.CREATE B:VENT2

ENTER RECORD STRUCTURE AS FOLLOWS:
FIELD       NAME,TYPE,WIDTH,DECIMAL PLACES
001         NRFACT,C,3
002         PRECIO,N,6
003
INPUT DATA NOW?
```

.(Aquí sigue la introducción de los datos de los dos registros...)

```
.USE B:VENT2
.LIST
```


UPDATE - Poner al día el archivo VENTAS

```
00001        111    300000
00002        166    200000

.USE B:VENTAS
.INDEX ON NRFACT TO B:INNRFACT
.USE B:INNRFACT INDEX NRFACT
UPDATE ON NRFACT FROM INNRFACT ADD PRECIO
.LIST
```

00001	Alfa A.	Calle Madrid, 5	Barcelona	111	1985/06/05	400000	12000	112000
00005	Alfa A.	Calle Madrid, 5	Barcelona	166	1985/09/25	700000	60000	560000
00002	Beta B.	Calle Avila, 5	Sevilla	333	1985/05/10	200000	24000	224000
00004	Delta D.	Calle Huesca, 5	Madrid	444	1985/07/20	400000	48000	448000
00003	Gamma C.	Calle Orense, 5	Madrid	222	1985/08/05	300000	36000	336000

Como se ve, el precio de la factura del primero y del segundo registros se han modificado de 100.000 a 400.000 pesetas y de 500.000 a 700.000 pesetas; los otros datos han permanecido invariables. (Con REPLACE en lugar de ADD se habría sustituido 100.000 por 300.000 y 500.000 por 200.000.)

JOIN - Unión del archivo VENTAS con otro archivo en un tercero

Finalmente, procedemos a la unión del archivo VENTAS con el archivo AGENDA en un tercer archivo denominado AGVENT y formado por los siguientes campos: NOMBRE, FECHA, OBLIGACIONES, CADUCIDAD e IVA.

```
.SELECT PRIMARY
.USE B:VENTAS
.SELECT SECONDARY
.USE B:AGENDA
.JOIN TO B:AGVENT FOR NOMBRE=S. NOMBRE FIELD
  NOMBRE,
FECOB, CADUC, IVA
.USE B:AGVENT
.LIST STRUCTURE

STRUCTURE FOR FILE: B:AGVENT
NUMBER OF RECORDS: 00005
DATE OF LAST UPDATE: 10/02/86
PRIMARY USE DATABASE
```

JOIN - Unión del archivo VENTAS con otro archivo en un tercero

FLD	NAME	TYPE	WIDTH	DEC
001	NOMBRE	C	08	
002	FECOB	C	10	
003	CADUC	C	10	
004	IVA	N	06	
TOTAL			00035	

.LIST

00001	Alfa A.	1985/03/15	Pag.IVA	12000
00002	Beta B.	1985/04/15	Pag.IVA	24000
00003	Gamma C.	1985/05/15	Pag.IVA	36000
00004	Delta D.	1985/06/15	Pag.IVA	48000
00005	Alfa A.	1985/07/15	Pag.IVA	60000

CAPITULO VIII

INFORMES, CALCULOS E IMPRESION

Creación de informes y tablas

El dBASE prevé la posibilidad de disponer informes y tablas a las que transferir los datos almacenados en el archivo.

REPORT es la orden que permite formatear los datos en líneas y columnas. Esto no sólo resulta útil para quien utiliza el dBASE en la elaboración de facturas o de la contabilidad, sino que también es útil para la presentación de los datos cuando éstos no ocupan mucho espacio. Por ejemplo, con REPORT se pueden hacer columnas con las direcciones o los nombres de los clientes que vivan en una determinada ciudad. No es necesario que el REPORT de un archivo tenga que ser total: en efecto, los datos contenidos en él se pueden elegir según las necesidades.

Cada vez que se activa REPORT automáticamente se crea un archivo de tipo FRM, que puede ponerse al día, imprimirse y visualizarse en cualquier momento.

Para el formateo de un archivo, después de abrirlo como siempre con USE, hay que llevar a cabo las siguientes operaciones:

- teclear REPORT para llamar a los programas con las órdenes del REPORT;
- escribir el nombre del REPORT para denominar el archivo;
- introducir los parámetros de impresión;
- pulsar "Y" o "N" para encabezar o no la página;
- escribir, si se optó por ponerlo, el encabezamiento de la página;
- pulsar "Y" o "N" para la separación de los campos;

- pulsar "Y" o "N" para los totales de los campos numéricos;
- introducir las dimensiones y el nombre del campo;
- introducir el encabezamiento de las columnas correspondientes al campo;
- pulsar <RETURN> para memorizar las especificaciones en el archivo de tipo ".FRM".

Especifiquemos esto un poco más: después de USE <nombre del archivo> se hará:

REPORT
ENTER REPORT FORM NAME

Para contestar al mensaje hay que escoger y escribir el nombre del archivo, por ejemplo REPNOME, que contiene los datos formateados;

REPNOME
ENTER OPTIONS, M = LEFT MARGIN, l = LINES/PAGE,
W = PAGE WIDTH

Este mensaje ofrece la posibilidad de determinar las dimensiones de la página según los datos que deban ser introducidos. Los valores estándar, que se utilizan en un formateado normal, son:

M	margen izquierdo:	— 8 caracteres
L	líneas por página:	— 56 líneas
W	caracteres por línea:	— 80 caracteres

Indicando, de una forma general, con nn los valores escogidos por nosotros, tendremos que teclear:

M = nn
L = nn
W = nn

o bien nos limitaremos a apretar <RETURN> aceptando así los valores estándar.

En este punto, el ordenador pregunta:

PAGE HEADING? (Y/N)

La respuesta al mensaje sirve para incluir o no un título en la página.

Contestamos "Y", por lo que aparece el mensaje

ENTER PAGE HEADING:

tras el cual hay que introducir el nombre del encabezamiento, seguido por <RETURN> como siempre. Entonces el dBASE pregunta:

DOUBLE SPACE RECORD? (Y/N)

Si se contesta "Y", las columnas que contienen los datos de los campos se separarán con un doble espacio entre ellas. Con "N" se separarán con un espacio normal.

Sigue la pregunta del ordenador:

ARE TOTALS REQUIRED? (Y/N)

La respuesta a esta pregunta sirve para disponer los totales de los campos que contienen datos numéricos.

Supongamos que contestamos afirmativamente (Y). Ahora la pregunta es:

SUBTOTALS IN REPORT? (Y/N)

La respuesta permite calcular los subtotales de los campos que contienen datos numéricos.

El ordenador escribirá después de todo esto en la pantalla lo siguiente:

COL   WIDTH, CONTENTS
001

Una vez aquí hay que completar la orden describiendo un campo tras otro.

¡Ojo! En un archivo puede haber muchísimos campos, pero no tienen por qué servir todos para la representación de un REPORT; por lo tanto, se pueden utilizar sólo los campos que sean necesarios y dejar los otros.

Con este espíritu teclearemos los diferentes "Dimensión, Nombre del campo" que nos interesen. El ordenador hará aparecer tras cada inserción el mensaje

ENTER HEADING:

El texto con que respondamos encabezará cada columna que contenga los datos del campo llamado. En el caso de que no sea necesario el encabezamiento será suficiente con dar <RETURN>.

Sigue la pregunta:

ARE TOTALS REQUIRED? (Y/N)

Con la respuesta a esta pregunta se determinan los totales de las columnas en el caso de que éstas contengan datos numéricos.

Una vez definidos los campos que interesan se da a la orden <RETURN>. El archivo, ya formateado, se memorizará en un archivo de tipo ".FRM". Después de una breve elaboración aparecerá el listado en la pantalla, tal y como se haya dispuesto.

Antes de empezar a trabajar con la orden REPORT sería aconsejable hacer una prueba para comprobar la posición exacta de los datos; esto evitará errores y, por lo tanto, tener que repetir todo el proceso.

Un archivo REPORT puede ser llamado en cualquier momento.

En el dBASE, si se cometen errores durante el planteamiento del REPORT, se pueden corregir con la orden MODIFY REPORT sin necesidad de reestructurarlo desde el principio. Además, en el dBASE III está prevista la orden LABEL para realizar impresiones específicas, como las etiquetas de las direcciones. El cuadro de impresión se memoriza en un archivo de tipo LBL y puede ser modificado con la orden MODIFY LABEL.

Impresión de un archivo formateado

El archivo formateado se puede imprimir efectuando las siguientes operaciones:

- escribir USE <nombre del archivo>;
- escribir REPORT FORM <nombre del archivo de report> TO PRINT, o bien REPORT FORM <nombre del archivo de report> FOR "expresión" TO PRINT para imprimir.

REPORT FORM <nombre del archivo de report> TO PRINT

Esta orden sirve para elaborar el archivo, después de lo cual el resultado se transmite a la impresora.

En el dBASE III la impresión de los archivos creados con LABEL se efectúa con la orden:

LABEL FORM <nombre del archivo LBL> (SAMPLE) ((Especificaciones del registro)) (FOR/WHILE condición) (TO PRINT) (TO FILE <nombre del archivo>)

Contar los registros de un archivo

Para poder saber cuántos registros de datos característicos hay en un archivo y con qué tipo de datos han sido completados, se puede utilizar la orden COUNT, con la siguiente sintaxis:

COUNT FOR <expresión>

La orden cuenta todos los registros que contienen la expresión especificada.

Una vez que se ha ejecutado la orden aparece el siguiente mensaje:

COUNT 000n

Indicando el número "n" de registros que cumplen la condición.

La condición puede tomar varias formas y tiene que estar bien especificada. Además puede usarse con las funciones de comparación lógicas (>, <, =, .AND., .NOT., .OR.) que ya hemos nombrado varias veces. Recordamos también que los datos alfanuméricos van entre comillas.

La sintaxis completa de este comando es:

COUNT [<alcance>] [FOR <expresión>] [TO <memoria variable>] [WHILE <expresión>]

Si no se especifica el alcance, dBASE asume que el comando es aplicable a todos los registros del archivo en "USE".

Sumas entre los campos numéricos (SUM)

Con el dBASE se puede efectuar también la suma de campos numéricos de un archivo utilizando la orden SUM. Las sumas pueden depender de condiciones que son establecidas por la orden misma.

La sintaxis es la siguiente:

SUM FIELD(S) <nombre del campo> FOR "Expresión"

Con esta orden se determina qué campos deben sumarse entre ellos y a qué condición tienen que responder tales datos. La formulación de la expresión FOR sigue la reglas recién vistas en el párrafo anterior.

En el dBASE III las operaciones "verticales" entre varios campos numéricos de un archivo se pueden efectuar también con la orden

AVERAGE <expresiones> (<especifica registro>)
(FOR/WHILE <δ=n>) TO <lista de las memorias variables>.

Impresión de los datos y del proceso

La impresión de los datos y todo lo que aparece en pantalla con el dBASE se realiza con SET PRINT ON. Una vez dada la orden SET PRINT ON todo lo que aparezca en la pantalla aparecerá en el papel.

Para anular la conexión entre impresora y ordenador es suficiente con teclear SET PRINT OFF.

He aquí una típica secuencia operativa:

- escribir USE <nombre del archivo a imprimir>;
- escribir SET PRINT ON para conectar la impresora al ordenador;
- escribir LIST o DISPLAY ALL para imprimir todos los datos del archivo en papel;
- escribir SET PRINT OFF para anular la conexión entre impresora y ordenador.

Visualización de los archivos guardados en disco

Efectuando las siguientes operaciones se pueden comprobar en la pantalla los archivos que están contenidos en un disco:

- teclear LIST FILES ON B:
  o bien
  teclear DISPLAY FILES ON B:
  para visualizar los archivos de tipo ".DBF" del directorio del disco B.
- teclear LIST FILES LIKE *.*
  o bien
  teclear DISPLAY FILE LIKE *.*
  para visualizar el directorio del disco A (donde está el dBASE).

Ordenes especiales (?, Reset, Eject)

Se trata de órdenes a utilizar en situaciones "especiales" que puedan crearse durante la elaboración del programa. Pongamos un ejemplo.

Cuando nos hayemos bloqueados y no sepamos seguir adelante, la solución es fácil: la(s) tecla(s) de RESET anulan todo y nos devuelven al sistema operativo. El RESET tiene que usarse con precaución, ya que anula todos los datos no memorizados.

Otra orden especial es ?. Sirve para efectuar operaciones matemáticas o bien para visualizar el contenido de un campo o de una variable. No hay que olvidar que en la programación se utiliza para posicionar cualquier variable y su contenido en la pantalla y en papel. Se puede hablar de una orden multiuso.

EJECT es una de las órdenes que se utilizan exclusivamente para la impresión. Permite que avance la página. La orden lee sólo las columnas y las líneas formateadas con el comando §

Ejercicios

Refiriéndonos siempre al archivo VENTAS, realicemos e imprimamos un informe que contenga los elementos siguientes: el nombre del cliente, la ciudad, la fecha, el número de la factura, el precio, el IVA y el total.

REPORT - Formación de una tabla para el archivo VENTAS

```
    .USE B:VENTAS
    .REPORT FORM B:VENTAS TO PRINT (facultativo)

ENTER OPTIONS M = LEFT MARGIN, L = LINES/PAGES,
  W = PAGE WIDTH
PAGE HEADING? (Y/N) Y
ENTER PAGE HEADING: INFORME DE LAS VENTAS
DOUBLE SPACE REPORT? (Y/N) N
ARE TOTALS REQUIRED? (Y/N) Y
SUBTOTALS IN REPORT? (Y/N) N
COL    WIDTH, CONTENTS
001          8,NOMBRE
ENTER HEADING: CLIENTE
002          11, CIUDAD
```

REPORT - Formación de una tabla para el archivo VENTAS

```
ENTER HEADING: CIUDAD
003          10, FECHA
ENTER HEADING: FECHA
004          3, NRFACT
ENTER HEADING: NF
005          6, PRECIO
ENTER HEADING: PRECIO
006          6, IVA
ENTER HEADING: IVA
ARE TOTALS REQUIRED? (Y/N) Y
007          6, TOTAL
ENTER HEADING: TOTAL
ARE TOTALS REQUIRED? (Y/N) Y
008
PAGE NO. 00001
10/02/86/
```

INFORME DE LAS VENTAS

CLIENTE	CIUDAD	FECHA	NF	PRECIO	IVA	TOTAL
Alfa A.	Barcelona	1985/06/05	111	100000	12000	112000
Beta B.	Sevilla	1985/05/10	333	200000	24000	224000
Gamma C.	Madrid	1985/08/05	222	300000	36000	336000
Delta D.	Madrid	1985/07/20	444	400000	48000	448000
Alfa A.	Barcelona	1985/09/25	166	500000	60000	560000
TOTAL				15000000	180000	1680000

SUM - Suma de los campos numéricos en el archivo VENTAS

```
.USE B:VENTAS
.SUM PRECIO
1500000

.SUM IVA
180000

.SUM TOTAL
1680000
```


SUM - Suma de los campos numéricos en el archivo VENTAS

```
.SUM PRECIO FOR CIUDAD="Barcelona"
600000

.SUM TOTAL FOR CIUDAD="Madrid"
784000
```

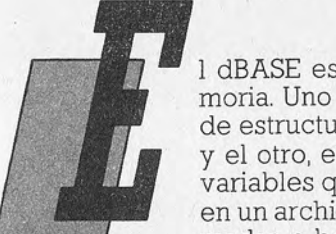
COUNT - Cuenta de los registros en el archivo VENTAS

```
.USE B:VENTAS
.COUNT
COUNT=00005
.COUNT FOR CLIENTE="Alfa A."
COUNT=00002
.COUNT FOR CIUDAD="Madrid"
COUNT=00002
.COUNT FOR IVA>30000
COUNT=00003
```

CAPITULO IX

LAS VARIABLES EN dBASE

Definición y asignación de valores



El dBASE está dotado de dos apartados de memoria. Uno se aprovecha para el trabajo normal de estructuración y elaboración de los archivos, y el otro, en cambio, se utiliza para manejar las variables que se crean. Estas últimas se recogen en un archivo de tipo MEM, y desde aquí son llamadas o borradas. En dicho archivo se pueden memorizar hasta 64 (dBASE II) ó 256 (dBASE III) variables y constantes, cada una de las cuales no puede tener una longitud superior a 254 caracteres.

Pero, ¿qué es una variable de memoria en el dBASE?

Como ocurre en cualquier otro lenguaje de programación, las variables son espacios de memoria, marcados con nombres a los que se pueden asignar valores. Para entender el concepto prueben a imaginarse unas cajas marcadas con un mismo nombre y que contienen valores diferentes cada vez. Para atribuir el nombre a una variable hay que tener en cuenta lo que sigue:

- el nombre de la variable no debe superar la longitud de 10 caracteres alfanuméricos;
- tiene que empezar necesariamente con un carácter alfanumérico. Los caracteres siguientes al primero pueden ser numéricos;
- en el nombre de la variable no se pueden utilizar signos de puntuación ni insertar espacios en blanco.

Los valores que pueden asignarse a las variables pueden ser de tres tipos:

- magnitudes numéricas: están constituidas por números positivos o negativos, enteros o decimales, de longitud no superior a 10 caracteres y por expresiones matemáticas reconducibles a un número;
- cadenas de caracteres: éstas se delimitan por comillas simples (') o dobles ("). Pueden contener caracteres alfanuméricos, valores matemáticos, caracteres especiales y espacios en blanco, siempre que tengan una longitud no superior a 254 caracteres. Por ejemplo, "Año 1985";
- valores lógicos: son los valores booleanos "T" por TRUE (verdadero) y "F" por FALSE (falso). Tienen una longitud de 1 carácter.

La creación de una variable se hace mediante la orden:

STORE <expresión> TO <nombre de la variable>

en la que:

- "expresión" indica un número, una expresión matemática, una cadena o un valor lógico;
- "nombre de la variable" indica el nombre asignado a la variable.

Normalmente al dar al <RETURN> el dBASE nos repite el valor asignado (si es una fórmula la calcula).

Para poder visualizar el contenido de la variable se utiliza el punto de interrogación seguido del nombre de la variable misma.

Además, la memorización de cadenas y de operaciones se puede efectuar también con las órdenes ACCEPT, INPUT y WAIT.

La primera sirve para memorizar cadenas alfanuméricas sin limitaciones de longitud y se activa según la sintaxis siguiente:

ACCEPT <"cadena de caracteres"> TO <nombre de la variable>

La segunda sirve para memorizar las cadenas numéricas o lógicas y se activa de la siguiente manera:

INPUT <"cadena numérica lógica"> TO <nombre de la variable>

Con la orden WAIT se pueden memorizar en las variables de memoria expresiones-clave muy útiles para elaborar las instrucciones de las funciones de un menú; esta orden se activa con la forma:

WAIT <expresión> TO <nombre de la variable>

Cómo conservar, comprobar y borrar variables

Después de haber constituido las variables, puede ser necesario conservarlas en un archivo adecuado, que asume la especificación MEM.

Esta operación es posible con la orden

SAVE TO <nombre del archivo de tipo MEM>

También es posible visualizar la lista de las variables y sus características (nombre, tipo, contenido) con la orden

DISPLAY MEMORY

Una vez que las variables ya no nos sirven se pueden borrar con las órdenes:

RELEASE ALL
Borra todas las variables

RELEASE NOMBRE VARIABLE
Anula sólo la variable especificada

Cuando RELEASE se ha utilizado solo, el dBASE crea espacio para insertar nuevas variables. No hay que confundir esta orden con ERASE, que sólo tiene función de "barrendero" de la pantalla.

Finalmente, con la orden

RESTORE FROM <nombre de la variable memorizada con SAVE>

es posible restablecer las variables anuladas llamándolas desde el archivo donde se han almacenado.

Veamos algún ejemplo de utilización de variables.

Ejemplos

Después de haber entrado en ambiente dBASE y de haber borrado la pantalla con la orden ERASE, supongamos que queremos calcular el área de un triángulo mediante la utilización de variables.

Como primer paso se asignan las variables:

- a la variable BASE se le atribuye el valor 10 con la orden:
  STORE 10 TO BASE
- a la variable altura el valor 5 con la orden:
  STORE 5 TO ALTURA

Ahora se pasa a asignar a otra variable, que será denominada AREA, la expresión matemática que calcula el área del triángulo. La asignación se hace de la siguiente manera:

```
STORE BASE*ALTURA/2 TO AREA
```

Después se comprueba el contenido de la variable AREA con la orden:

```
? AREA
```

en la línea de abajo se lee:

```
25
```

que representa, precisamente, el valor del área del triángulo, verificable fácilmente mediante un veloz cálculo mental.

Ahora comprobamos lo que está contenido en memoria después de haber realizado las operaciones anteriores. Para hacerlo basta con dar a la orden DISPLAY MEMORY:

```
BASE      (N)  10
ALTURA    (N)  5
AREA      (N)  25

**TOTAL**  03 VARIABLES USED 00021 BYTES USED
```


CAPITULO X

PROGRAMAR EN dBASE

Características fundamentales

Hasta ahora el dBASE ha sido presentado como un sistema que permite desarrollar las distintas operaciones por medio de la activación de variadas órdenes. Para resolver problemas complejos, sin embargo, estas órdenes pueden ser utilizadas a la vez y según un orden determinado, dando así origen a un auténtico *programa.*

El programa no es, por tanto, más que la descripción de los datos y las órdenes que hay que llevar a cabo, en una secuencia determinada, con el fin de desarrollar un trabajo en concreto.

Los datos del programa pueden ser de dos tipos:

- sencillos si están formados por datos únicos (por ejemplo, un número, una letra, una palabra, etc.);
- estructurados si están constituidos por un conjunto organizado de datos (por ejemplo, tablas).

Las órdenes del programa sirven, principalmente, para transmitir datos desde el exterior hacia el ordenador, para realizar determinadas manipulaciones sobre los datos introducidos en el ordenador y para transmitir los datos, elaborados o no, desde el ordenador hacia el exterior.

Para pasar desde el programa propuesto a la redacción definitiva del programa puede ser interesante utilizar el método de la pseudocodificación, que consta de las siguientes fases:

- definición del problema, eventual descomposición en subprogramas e individualización de los datos y de los resultados del problema; es una fase delicada e importante que requiere mucha atención;
- especificación de los recursos a utilizar, es decir, de los archivos de introducción de los datos, de los archivos de trabajo, de los archivos de resultados, de las variables, de las constantes, de los operadores matemáticos, etc.;
- descripción de las operaciones a cumplir para resolver el problema. Se deben especificar las operaciones de adquisición de datos del exterior, las operaciones de elaboración a cumplir, operaciones de salida de los resultados al exterior, etc.

Las diferentes fases de este método están generalmente descritas en una hoja de pseudocodificación, donde están especificados el nombre del programa, las asignaciones de los recursos, la descripción de las operaciones a desarrollar y la indicación de las órdenes dBASE a utilizar.

Después de la realización de esta hoja se pasa al programa en lenguaje dBASE.

Los programas de dBASE se basan sobre cuatro estructuras fundamentales:

- la estructura secuencial, constituida por una secuencia de órdenes y de instrucciones (Fig. 1);
- la estructura iterativa, que contiene las instrucciones de repetición DO WHILE...ENDDO (Fig. 2);
- la estructura alternativa, que contiene instrucciones alternativas del tipo IF...ELSE...ENDIF (Figs. 3 y 4);
- la estructura de procedimiento, que contiene llamadas a subarchivos que, a su vez, liberan una función particular (Fig. 5).

En los programas dBASE las órdenes se insertan sin ningún formato especial, y pueden contener:

- líneas de asignación de valores a variables (de una constante, de un resultado o de una expresión);
- líneas de expresiones, es decir, de constantes, variables, de operadores, de funciones, de órdenes;
- líneas de comentarios, para indicar saltos, elecciones, repeticiones y otras informaciones útiles.

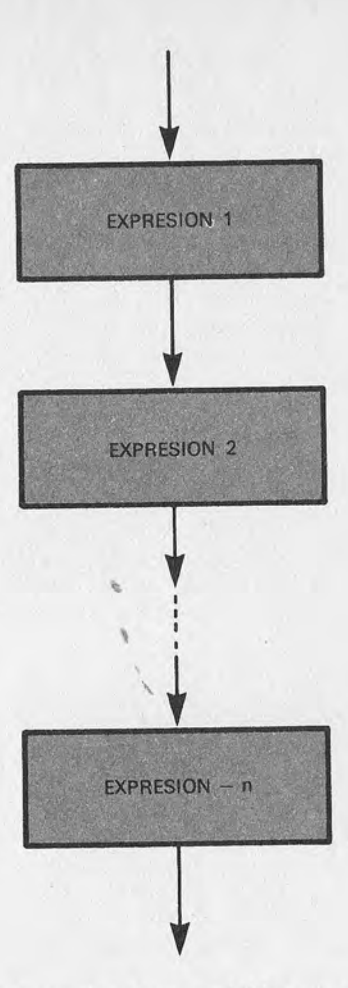


Figura 1.—En la estructura secuencial las expresiones del programa se ejecutan una después de la otra, según el orden en que se han escrito.

Escritura y comienzo de un programa

Después de entrar en ambiente dBASE, para escribir los programas se utiliza la orden

MODIFY COMMAND <nombre del archivo>

en donde el "nombre del archivo" es el que hay que atribuir al

programa mismo. El dBASE continúa con el mensaje NEW FILE y muestra una pantalla perfectamente limpia sobre la que se pueden empezar a transcribir las instrucciones del programa. Los programas se memorizan en un archivo de órdenes que está marcado con PRG si está compilado bajo el sistema operativo MS/DOS y con COM si está compilado bajo el sistema operativo CP/M.

Cada vez que se memoriza un archivo de programa, el dBASE automáticamente produce una copia de seguridad (back-up) marcada con la extensión BAK.

Para guardar el programa y volver al "." basta pulsar CTRL+W.

Para ejecutar el programa hay que escribir

DO <nombre del programa>

Esta orden abre el archivo de órdenes donde está contenido el programa y lo ejecuta de forma secuencial.

Dado que lo normal es mantener el disco con el dBASE en la unidad "A" y el disco de datos en la "B", lo más común es que el programa deba ir precedido con "B:", como verá en los ejemplos.

Procedimientos iterativos

Una de las ventajas del ordenador es la de poder repetir indefinidamente la misma secuencia sin cansarse.

La repetición de acciones siempre iguales se llama ciclo iterativo o repetitivo (Fig. 2). Se activa con la orden

```
DO WHILE condición
    instrucciones
ENDDO
```

En cuanto se llega al ciclo se valora la condición y entonces:

- Si resulta VERDADERA se ejecutan las instrucciones indicadas en las líneas siguientes hasta llegar a la orden ENDDO, la cual envía la ejecución del programa a la orden inicial de DO WHILE;
- Si la condición resulta FALSA, las instrucciones especificadas no se ejecutan y el programa continúa con la ejecución de las instrucciones que siguen a la orden ENDDO.

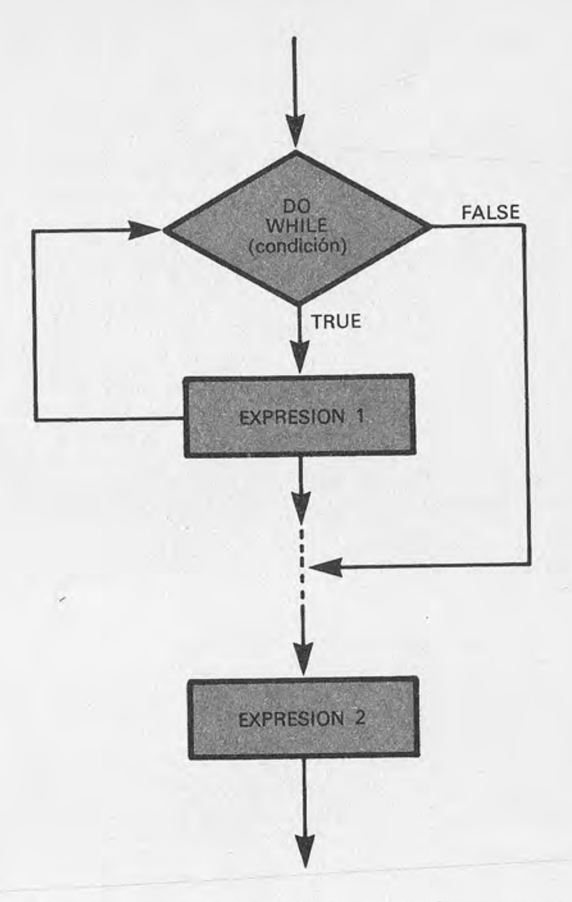


Figura 2.—La expresión DO WHILE...ENDDO corresponde al siguiente concepto: "Al verificarse la condición TRUE, se ejecuta la expresión 1 y se vuelve a la orden inicial de DO WHILE; si se verifica la condición FALSE se salta la expresión 1, pasando a la expresión 2".

Procedimientos alternativos

Cuando nos encontramos frente a dos o más vías a seguir, en el sistema dBASE la elección puede efectuarse con la pareja de órdenes IF/ENDIF, cuya sintaxis es la siguiente (Figs. 3 y 4):

```
IF condición
    instrucciones
ENDIF
```

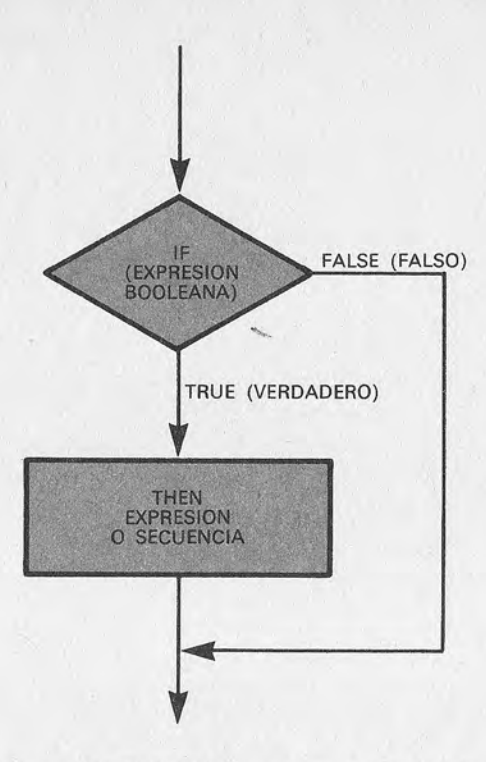


Figura 3.—La expresión IF...THEN hace que, al ser cierta la expresión booleana (TRUE), se ejecute una determinada expresión o secuencia; si, en cambio, la expresión booleana no se verifica, esta secuencia se salta.

La orden permite seguir recorridos alternativos según se verifique o no la condición especificada. Funciona como sigue: apenas la orden IF abre el ciclo, se valora la condición y entonces:

- si es VERDADERA se ejecutan las instrucciones especificadas hasta la orden ENDIF, para proseguir después con las instrucciones puestas a continuación de la última orden;
- si la condición resulta FALSA, entonces el dBASE no ejecuta las instrucciones que siguen al IF y salta a las que están puestas después del ENDIF.

Cuando se tienen diferentes caminos a seguir es aconsejable utilizar, en lugar de IF/ENDIF, la pareja de órdenes DO CASE/ENDCASE, cuya sintaxis se presenta como sigue (Fig. 5):

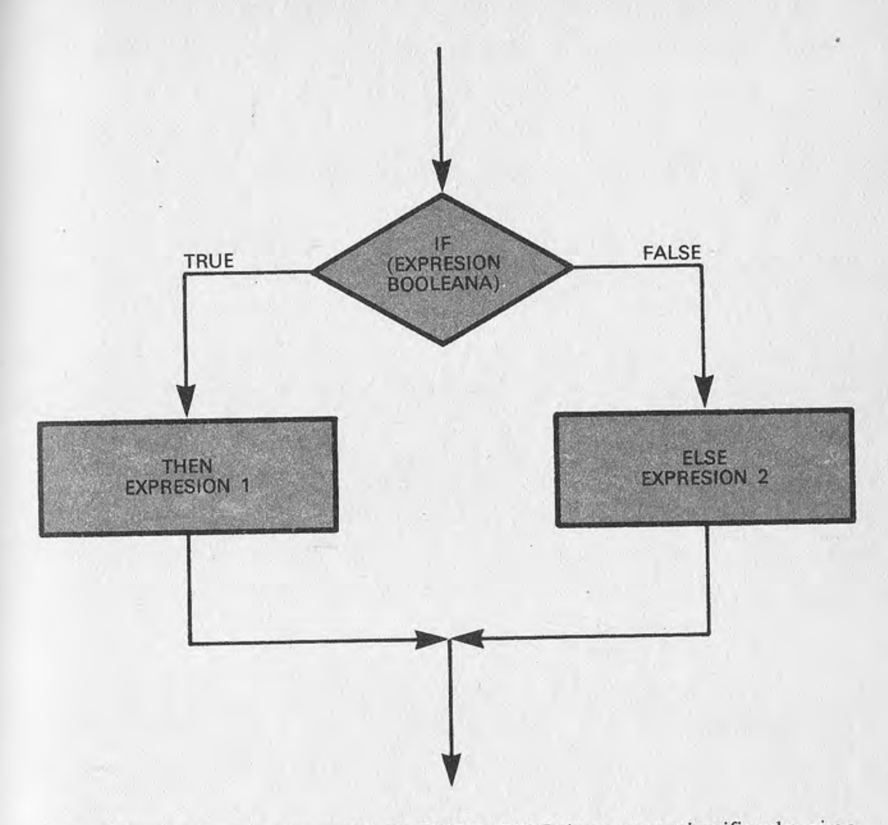


Figura 4.—La expresión IF...THEN...ELSE tiene este significado: si se verifica una determinada expresión booleana (TRUE) entonces se ejecuta la expresión 1; si no se verifica (FALSE) se ejecuta la expresión 2.

```
DO CASE condición
    instrucciones
ENDCASE
```

Esta orden se utiliza bastante para los menús, ya que el usuario tiene posibilidad de actuar de varias maneras.

Apenas la orden abre el ciclo, se valora la condición y:

- si esta condición se satisface, se ejecutan todas las instrucciones establecidas hasta el siguiente CASE, y así sucesivamente hasta el ENDCASE;
- si la condición no se satisface para ninguno de los casos, salta a las instrucciones que siguen a la estructura CASE.

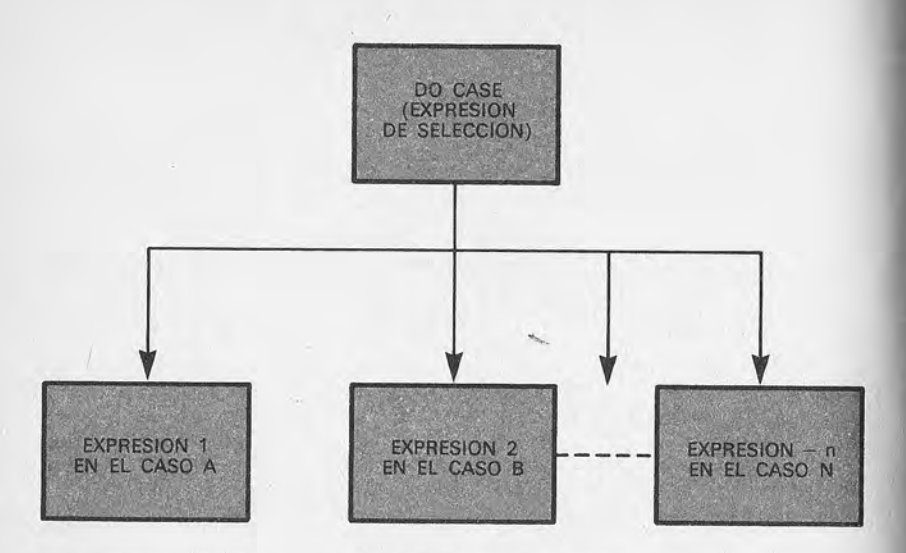


Figura 5.—La expresión DO CASE...ENDCASE encierra varias alternativas: Cuando se verifique el caso "A", "B" o "N" se ejecuta la expresión correspondiente a "A" o, respectivamente, "B" o "N". Si no se verifica ninguno de los casos enumerados, el programa continúa adelante.

Colocación de los datos

La pantalla del ordenador se puede comparar a una pizarra en la que se pueden colocar a voluntad palabras, números y caracteres.

En general, las pantallas de ordenadores tienen 24 líneas y 80 columnas. Para poder colocar correctamente los datos en la pantalla hay que especificar las coordenadas de colocación, que están formadas por el número de línea y el número de columna.

Las 24 líneas de la pantalla están numeradas del 0 al 23: la primera línea es el 0 y la vigésimo cuarta es la 23.

Las 80 columnas están numeradas del 0 al 79: la primera es la columna 0, mientras que la última es la columna 79.

La línea 0 se utiliza para visualizar eventuales mensajes, por lo que es necesario dejarla vacía siempre.

La especificación de las coordenadas de la pantalla se activa con la siguiente orden:

@ <coordenadas> SAY 'expresión'

- coordenadas: son los números de las coordenadas (línea y columna) separados entre ellos por una coma. Indican la posición en la que tiene que empezar la visualización;
- expresión: puede ser el nombre de una variable, un número, una cadena o una expresión matemática.

Con el dBASE se puede establecer el formato de impresión de cualquier variable. Por ejemplo, si se tuviese que visualizar una fecha en el formato "barrado", sería suficiente con indicar:

```
ERASE
STORE '          ' TO MFECHA
@20,10 SAY 'FECHA' MFECHA PICTURE'99/99/99'
READ
@22,10 SAY MFECHA
```

De esta forma pedirá la fecha en la posición 20,10 (línea 20, columna 10) en el siguiente formato:

```
FECHA:  /  /  :
```

y pondrá la que le demos en la 22,10 como: 12/11/85.

Se puede establecer también el formato de impresión de las cifras numéricas, utilizando conjuntamente el símbolo # (llamado cancela), con la opción USING.

La sintaxis de esta opción es la siguiente:

USING <# #,œ œ œ,# # #>

donde los símbolos # se usan para indicar la posición que deberán tomar las cifras en el momento de la impresión.

Por ejemplo, para visualizar la cifra 20.000.000 se utilizará el siguiente formato:

```
ERASE
STORE 20000000 TO CAPITAL
     20,10 SAY 'CAPITAL'
     22,10 SAY CAPITAL USING "# #, # # #, # # #"
```

En la pantalla aparecerá:

20,000,000

La utilización de esta opción USING alarga en tres posiciones la impresión de las variables.

Introducción interactiva de datos

La introducción interactiva de datos se puede realizar a través de las dos fases que aquí se describen.

En la primera se crean variables vacías con valor nulo: espacios para las cadenas y valores "0" para los números. Esta fase se activa con la orden

STORE TO

En la segunda se pasa a la introducción en las variables vacías de los valores de letras (cadenas) y números correspondientes a los datos introducidos. Esta fase se realiza con la combinación de órdenes:

@<coordenadas> SAY <expresión> GET <variable> READ

La primera parte sirve para visualizar en la pantalla, a partir del punto que tiene las coordenadas especificadas, la expresión establecida y el contenido de la variable asignada. La segunda orden (READ) sirve para leer el valor de la variable.

Ejercicios

Programa para el cálculo del área del triángulo.

Presentamos ahora un programa que permite calcular el área de un triángulo (los "*" identifican los comentarios).

PROCEDIMIENTO - AREA DEL TRIANGULO

```
* Definición del ambiente operativo

RELEASE ALL
SET TALK OFF
ERASE
*

* Creación de las variables vacías
STORE "      "TO base
STORE "      "TO altura
*
```

PROCEDIMIENTO - AREA DEL TRIANGULO

```
* Creación de la plantilla e introducción interactiva de los datos
      @6,10 SAY'CALCULO AREA DEL TRIANGULO'
      @7,10 SAY'-------------------------------------------'
      @9,10 SAY' Longitud de la base.............'GET base
      @10,10 SAY' Longitud de la altura............'GET altura
READ
*

* Transformación de las variables de cadena en numéricas

STORE VAL(base) TO b
STORE VAL(altura) TO a
*

* Cálculo del área del triángulo y su visualización

STORE b*a/2 TO r
@12,10 SAY'     Area del triangulo'GET r
```

EJEMPLO - PROGRAMA AREA DEL TRIANGULO

```
     CALCULO AREA DEL TRIANGULO

Longitud de la base..........................................: 10 :
Longitud de la altura........................................: 3  :

Area del triángulo: 15          :
```

Programa para la elaboración del archivo BIBLIO

El siguiente programa permite la elaboración del archivo BIBLIOTECA, ya estructurado con CREATE, por medio de un menú que prevé las siguientes elecciones: PUESTA AL DIA, LISTA, CLASIFICACION, BUSQUEDA. Para cada elección se presentará su correspondiente procedimiento.

- PUESTA AL DIA. Seleccionando el número 1 aparece la plantilla de APPEND para el procedimiento de puesta al día del archivo. Esta contiene todas las funciones normales de

edición; para volver al menú principal es suficiente con dar a la tecla CTRL-Q en un registro vacío;

- LISTA. Introduciendo el número 2 aparece la lista de los libros, completada con todos los datos incluidos a través de la función de puesta al día;
- CLASIFICACION. Pulsando el número 3 se procede a la clasificación automática del archivo según el nombre del autor, por ejemplo. La elección del campo clave de clasificación depende de las necesidades del usuario. El archivo que contiene los datos ordenados también se puede listar con los campos TITULO, EDITORIAL, AÑO DE EDICION.
- BUSQUEDA. Con el número 4 se procede a la búsqueda de datos. La programación de búsqueda se desarrolla en dos fases.

La primera consiste en memorizar las variables que después se utilizarán para la comprobación durante la búsqueda. Si la variable resulta vacía, se vuelve al menú principal; si no, se procede a la búsqueda mediante la subcadena. Esto se hace mediante la orden LOCATE y la función "&", que tiene el cometido de leer el contenido de la variable. También se ha previsto el caso de que el libro buscado no exista en la biblioteca. Esto ocurre si se verifican dos condiciones: si el puntero llega al final del archivo (EOF) sin haber encontrado nada, o bien (.OR.) si el número del registro se vuelve 0 (#=0). Todo esto conlleva la visualización del mensaje NO ENCONTRADO.

La segunda parte prevé la composición de la "FICHA INFORMATIVA DEL LIBRO", en la que se colocan todos los datos referentes al volumen buscado: el autor, el título, el editor, el año de edición y el código. Además de esto, también se visualiza el número de los registros.

Para salir de la plantilla-menú basta con apretar la tecla ESC.

El listado contiene los comentarios de las órdenes y subrutinas, que se señalan con un asterisco.

PROGRAMA "MENU PRINCIPAL" DEL ARCHIVO BIBLIO

Como primer trabajo planteamos el procedimiento del menú principal que prevé las cuatro elecciones especificadas.

```
* Inicio del programa

ERASE
SET TALK OFF
```


PROGRAMA "MENU PRINCIPAL" DEL ARCHIVO BIBLIO

```
* Instalación de la subrutina de petición de elección de la opción.
STORE T TO MENU
DO WHILE T
ERASE

* Definición de la pantalla
    @ 5,10 SAY' ELABORACION ARCHIVO BIBLIOTECA'
    @ 6,10 SAY' ----------------------------------------------'
    @ 8,10 SAY'     1. Puesta al día'
    @ 9,10 SAY'     2. Lista'
    @10,10 SAY'     3. Clasificación'
    @11,10 SAY'     4. Búsqueda'
    @12,10 SAY'     <ESC> para salir'

* Memorización de la respuesta para la posterior comprobación
WAIT TO RESPUESTA

* Instalación del procedimiento de elección

DO CASE
```

EJEMPLO - MENU PARA LA ELABORACION DEL ARCHIVO BIBLIO

```
              ELABORACION ARCHIVO BIBLIOTECA
                1. Puesta al día
                2. Lista
                3. Clasificación
                4. Búsqueda
                <ESC> para salir
WAITING
```

PROCEDIMIENTO PARA LA PUESTA AL DIA DEL ARCHIVO BIBLIO

```
* Acción para el procedimiento de puesta al día

CASE RESPUESTA='1'
USE B:BIBLIO
APPEND
```

EJEMPLO DE PUESTA AL DIA DEL ARCHIVO BIBLIO

```
RECORD 00006
AUTOR     :          :
TITULO    :                :
EDITORIAL :           :
AÑOED     :    :
CODIGO    :    :
RESUMEN   :
```

PROCEDIMIENTO PARA LA LISTA DEL ARCHIVO BIBLIO

```
* Acción para el procedimiento de lista de los datos

CASE RESPUESTA='2'
ERASE
USE B:BIBLIO
LIST
?
?'Apretar <RETURN> para volver al menú principal'

WAIT
DO B:BIBLIO
```

EJEMPLO DE LA LISTA DE LOS LIBROS DEL ARCHIVO

```
00001 M. Banaham    UNIX. Guía profesional  Díaz de Santos
1985 A00001
00002 J. Sachs      El IBM PC               McGraw-Hill
1985 A00002
00003 A. Simpson    El libro del Lotus 1-2-3  Anaya Multime-
                                              dia
1985 B0010
00004 G. Clavel     Introducción a la pro-
                    gramación               Masson
1985 B006
00005 A. Nania      Diccionario de informá-
                    tica                    Paraninfo
1985 B002

Apretar <RETURN> para volver al menú principal.

WAITING
```


PROCEDIMIENTO PARA LA CLASIFICACION DEL ARCHIVO BIBLIO

```
* Acción para el procedimiento de clasificación de los datos
CASE RESPUESTA='3'
ERASE
USE B:BIBLIO
SORT ON AUTOR TO B:$AUTOR
USE B:$AUTOR
@2,10 SAY'LISTA DE LA BIBLIOTECA POR ORDEN ALFABETI-
        CO DE AUTOR'
@3,10 SAY'----------------------------------------'
LIST OFF AUTOR, TITULO, EDITORIAL, AÑOED
?
?' <RETURN> para volver al menú principal'
WAIT
DO B:BIBLIO
```

EJEMPLO DE LA LISTA DE LOS LIBROS POR ORDEN ALFABETICO DE AUTOR

```
LISTA DE LA BIBLIOTECA POR ORDEN ALFABETICO DE AU-
TOR

A. Nania     Diccionario de Informá-
             tica                      Paraninfo        1985
A. Simpson   El libro del Lotus 1-2-3  Anaya  Multime-
                                       dia              1985
G. Clavel    Introducción a la pro-
             gramación                 Masson           1985
J. Sachs     El IBM PC                 McGraw-Hill      1985
M. Banaham   UNIX. Guía profesional    Díaz de Santos   1985
<RETURN> para volver al menú principal

WAITING
```

PROCEDIMIENTO PARA LA BUSQUEDA EN EL ARCHIVO BIBLIO

```
* Acción para el procedimiento de búsqueda de datos

CASE RESPUESTA='4'
ERASE
```

PROCEDIMIENTO PARA LA BUSQUEDA EN EL ARCHIVO BIBLIO

```
USE B:BIBLIO
?
?
```

* Memorización de la variable MTITULO que contiene el título, entero o sólo una parte de él.

```
ACCEPT 'Título (completo o parcial) del libro'
TO MTITULO
?
```

* Memorización de la variable MEDITORIAL que contiene el nombre correspondiente de la editorial.

```
ACCEPT 'Editorial' TO MEDITORIAL
```

* Ordenes de búsqueda con uso de la subcadena y de la conjunción lógica AND.

```
LOCATE FOR '&MTITULO' $TITULO .AND. '&MEDITORIAL'
&EDITORIAL
```

* Caso en que la variable resulta vacía

```
IF MTITULO =' '
DO B:BIBLIO
ENDIF
```

* Caso en que los datos pedidos no existen; colocación en la pantalla del mensaje "no encontrado" y volver al principio del programa.

Si el puntero llega al final del archivo (EOF) o bien si (.OR.) el número del registro que contiene los datos pedidos resulta ser 0...

```
IF (EOF .OR. #=0).
```

Entonces en la pantalla aparecerá el siguiente mensaje:

```
ERASE
```


PROCEDIMIENTO PARA LA BUSQUEDA EN EL ARCHIVO BIBLIO

```
?'              ___________________________________________'
?'              EL LIBRO NO EXISTE EN NUESTRA BIBLIOTECA'
?'                   Pulse RETURN y vuelva a escribir'
?'                   el título del libro'                  '
                ___________________________________________
WAIT
DO B:BIBLIO
ENDIF
ERASE
```

* Si el libro es encontrado en el archivo, el número de registro se transforma en cadena y se memoriza en la variable MNUMERO.

```
ERASE
STORE STR (#,5) TO MNUMERO
ERASE
```

* Formateo de la pantalla para la composición de la plantilla que contiene los datos del libro.

```
?' ** FICHA INFORMATIVA DEL LIBRO **'
?' ========================================'
@4,5 SAY 'Número del registro:'
@4,58 SAY MNUMERO
@6,5 SAY 'Autor:'
@6,13 SAY AUTOR
@8,5 SAY 'Título:'
@8,13 SAY TITULO
@10,5 SAY 'Editorial:'
@10,13 SAY EDITORIAL
@10,59 SAY 'Año edición:'
@10,59 SAY AÑOED
@12,5 SAY 'Código:'
@12,16 SAY CODIGO
?'=========================================='
?
```

* Mensaje de vuelta al menú principal mediante la tecla de RETURN.

```
?' <RETURN> para volver al menú principal'
WAIT
```

PROCEDIMIENTO PARA LA BUSQUEDA EN EL ARCHIVO BIBLIO

* Llamamiento al programa que contiene el menú principal.

DO B:BIBLIO
<RETURN> para volver al menú principal.
WAITING

* Orden de cierre del programa con la clasificación de las variables de trabajo.

RELEASE ALL
ENDCASE
ENDDO
RETURN

EJEMPLO DE LA BUSQUEDA DE LIBROS EN EL ARCHIVO BIBLIO

FASE I: Introducción del título y de la editorial del libro.
Título (completo o parcial) del libro: PC.
Editorial: McGraw-Hill.
FASE II: Visualización de la ficha informativa del libro.

** FICHA INFORMATIVA DEL LIBRO **
==

Número del registro: 2

Autor: J. Sachs.
Título: El IBM-PC
Editorial: McGraw-Hill — Año de edición: 1985
Código: A0002

PROCEDIMIENTO PARA LA LISTA DEL ARCHIVO VENTAS

Supongamos que queremos efectuar la lista automática del archivo VENTAS. Se puede dar curso al procedimiento siguiente.

* Definición del ambiente de trabajo.
ERASE
SET TALK OFF

PROCEDIMIENTO PARA LA LISTA DEL ARCHIVO VENTAS

* Apertura del archivo VENTAS
USE B:VENTAS

* Clasificación según el nombre del cliente.
SORT ON NOMBRE TO B:NOMBRE
USE B:NOMBRE

* Lista de los clientes por orden alfabético.
@10,5 SAY' LISTA DE CLIENTES '
DISPLAY ALL OFF
WAIT

EJEMPLO DE LA LISTA DE CLIENTES

LISTA DE CLIENTES

Alfa A.	Calle Madrid, 5	Barcelona	111	1985/05/06	100000	0	0
Beta B.	Calle Avila, 5	Sevilla	333	1985/05/10	200000	0	0
Gamma C.	Calle Orense, 5	Madrid	222	1985/08/05	300000	0	0
Delta D.	Calle Huesca, 5	Madrid	444	1985/07/20	400000	0	0
Alfa A.	Calle Madrid, 5	Barcelona	166	1985/09/25	500000	0	0

PROCEDIMIENTO PARA EL CALCULO DEL IVA Y DEL IMPORTE TOTAL EN EL ARCHIVO VENTAS

Suponiendo que queramos calcular el importe del IVA y del total correspondiente, se puede seguir el procedimiento siguiente:

* Definición del ambiente de trabajo
ERASE
SET TALK OFF

* Apertura del archivo
USE B:VENTAS

* Cálculo del porcentaje del IVA
REPLACE ALL IVA WITH PRECIO*12/100

* Cálculo del importe total
REPLACE ALL TOTAL WITH PRECIO+IVA

PROCEDIMIENTO PARA EL CALCULO DEL IVA Y DEL IMPORTE TOTAL EN EL ARCHIVO VENTAS

```
* Lista parcial del archivo completado con los cálculos
  @10,5 SAY'**LISTA DE LAS FACTURAS**'
  USE B:VENTAS
  DISPLAY ALL OFF NOMBRE, CIUDAD, NRFACT, FEFACT,
     PRECIO, IVA, TOTAL
  WAIT
```

EJEMPLO DEL PROCEDIMIENTO DEL CALCULO DEL IVA Y EL IMPORTE TOTAL

** LISTA DE LAS FACTURAS **

Alfa A.	Calle Madrid, 5	Barcelona	111	1985/06/05	100000	12000	112000
Beta B.	Calle Avila, 5	Sevilla	333	1985/05/10	200000	24000	224000
Gamma C.	Calle Orense, 5	Madrid	222	1985/07/20	300000	36000	336000
Delta D.	Calle Huesca, 5	Madrid	444	1985/07/20	400000	48000	448000
Alfa A.	Calle Madrid. 5	Barcelona	166	1985/09/25	500000	60000	560000

APENDICE A

Con comparación entre el dBASE II y el dBASE III.

Tipos de datos aceptados

TIPO DE LOS DATOS	DESCRIPCION	dBASE II	dBASE III
Datos de tipo C	Todos los caracteres ASCII, comprendidos letras, números, símbolos y espacios.	SI	SI
Datos de tipo N	Todos los números positivos y negativos, los signos y cifras decimales.	SI	SI
Datos de tipo L	Todos los valores de tipo verdadero (Y, y, T, t) o falso (N, n, F, f).	SI	SI
Datos de tipo D	Fechas expresadas de forma estándar (mes/día/año).	NO	SI
Datos de tipo M	Campos Memo hasta 4000 caracteres de longitud.	NO	SI

APENDICE B

Operadores

Los operadores de dBASE II y III son de tipo lógico y de cadena. Si en una expresión se utilizan varios operadores lógicos, el orden de precedencia es: matemática, relacional y, después, lógica. Todas las operaciones del mismo nivel de precedencia se ejecutan en orden de lectura, de izquierda a derecha. Los paréntesis se utilizan para variar el orden en el que deben ser ejecutadas.

OPERADORES	DESCRIPCION	dBASE II	dBASE III
Operadores lógicos			
.NOT.	Negación lógica	SI	SI
.AND.	Conjunción lógica	SI	SI
.OR.	Disyunción lógica	SI	SI
Operadores de cadena			
+	Concatenación de cadenas	SI	SI
$	Búsqueda de subcadenas	SI	SI
Operadores matemáticos			
+	Suma	SI	SI
–	Resta	SI	SI
*	Multiplicación	SI	SI
/	División	SI	SI
**, ^	Exponencial	NO	SI
( )	Paréntesis de reagrupamiento	SI	SI
Operadores relacionales			
<	Menor que...	SI	SI
>	Mayor que...	SI	SI
=	Igual	SI	SI
<>, #	Distinto que...	SI	SI
<=	Menor o igual que...	SI	SI
>=	Mayor o igual que...	SI	SI

APENDICE C

Funciones

El dBASE (versión II y III) está provisto de funciones particulares que desarrollan utilidades y operaciones determinadas. Estas son:

FUNCIONES	SIGNIFICADO	dBASE II	dBASE III
Funciones para la fecha y la hora			
CDOW(exp)	Día de la semana en caracteres	NO	SI
CMONTH(exp)	Mes del año en caracteres	NO	SI
CTOD(exp)	Conversión de caracteres a datos	NO	SI
DATE ( )	Instalación de la fecha	SI	SI
DAY(exp)	Día del mes	NO	SI
DOW(exp)	Día de la semana en cifras	NO	SI
DTOC(exp)	Conversión de la fecha en caracteres	NO	SI
MONTH(exp)	Mes del año en cifras	NO	SI
TIME( )	Instalación de la hora	NO	SI
YEAR(exp)	Año expresado en cifras	NO	SI

FUNCIONES	SIGNIFICADO	dBASE II	dBASE III
Funciones para caracteres y cadenas			
&<var>	Lectura del contenido de la variable	SI	SI
?	Visualización del contenido de la variable	SI	SI
$	Extracción de una subcadena	SI	NO
!	Conversión de los caracteres en mayúsculas	SI	SI
*	Marca de los registros que son borrados	SI	SI
#	Lectura del número del registro	SI	NO
AT	Búsqueda de la subcadena	SI	SI
LEN	Medida de la longitud de la cadena	SI	SI
SPACE(expN)	Generación de espacios vacíos	NO	SI
SUBSTR <expc><start> <lenght>	Extracción de una subcadena	NO	SI
TRIM(expC)	Cancelación de espacios vacíos	SI	SI
TYPE(expC)	Visualiza el tipo de variable	SI	SI
Funciones para la conversión de caracteres			
ASC(expC) (expC2)	Conversión desde caracteres a código ASCII	SI	SI
CHR( )	Conversión desde código ASCII a caracteres	SI	SI
LOWER(expC)	Conversión en minúsculas	NO	SI
STR(expN)	Conversión desde números a caracteres	SI	SI
UPPER(expC)	Conversión en mayúsculas	NO	SI
VAL(expC)	Conversión desde caracteres a números)	SI	SI

FUNCIONES	SIGNIFICADO	dBASE II	dBASE III
Funciones para archivos y registros			
BOF( )	Inicio del archivo	NO	SI
DELETED	Registro borrado	NO	SI
EOF( )	Final del archivo	SI	SI
FILE(Filename)	Existencia del archivo	SI	SI
RECNO( )	Número del registro	NO	SI
Funciones matemáticas			
EXP(expN)	Exponencial	NO	SI
INT(expN)	Parte entera	SI	SI
LOG(expN)	Logaritmo natural	NO	SI
ROUND	Redondeo de los decimales	NO	SI
SQRT(expN)	Raíz cuadrada	NO	SI
Funciones para pantalla e impresión			
COL( )	Posición de las columnas en la pantalla	NO	SI
PCOL( )	Posición de la columna de impresión	NO	SI
PROW( )	Posición de la línea de impresión	NO	SI
ROW( )	Posición de la línea en la pantalla	NO	SI

APENDICE D

Sintaxis de los comandos órdenes por orden alfabético

La terminología utilizada en la sintaxis de los comandos ha sido:

<ALIAS >	= Nombre del archivo alternativo.
<CONDICION>	= Expresión lógica de la que depende la verificación de una acción.
<DEC>	= Número de las cifras decimales.
<DELIMITADOS>	= Comillas o paréntesis cuadrados que delimitan los datos.
<EXP><EXPRESION>	= Expresión representada por campos, variables de memoria, funciones, cadenas de caracteres o sus combinaciones.
<EXP C>	= Expresión de caracteres.
<EXP D>	= Expresión de datos.
<EXP N>	= Expresión numérica.
<FIELD> o <FIELDNAME>	= Nombre de un campo.
<FILE> o <FILENAME>	= Nombre de un archivo.
<INDEX FILE>	= Nombre de un archivo indexado (catalogado).
<KEY>	= Clave, entendida como la parte del registro utilizada para crear un archivo catalogado (indexado).
<LISTA EXP>	= Lista de expresiones separadas por una coma.
<LISTA FIELD>	= Lista de campos separados por una coma.
<LISTA VAR>	= Lista de variables de memoria separada por una coma.

<MASCARA> = Utilización de "*" o bien "?" en lugar de un nombre.
<n> = Un número.
<PROCEDURE> = Subprograma insertado en un archivo de órdenes (.PRG).
<RANGO> = Esfera de acción de todos o de "n" registros, o sólo del registro "n".
<STRING> = Cadena de caracteres.
<VAR> = Nombre de una variable de memoria.

SINTAXIS	dBASE II	dBASE III
?<exp> Muestra una expresión en la línea siguiente	SI	SI
?? Visualiza una expresión en la misma línea	SI	SI
@(coordenadas) (SAY<exp> (PICTURE<clause>)) (GET<var> (PICTURE<clause>) (RANGE<exp,exp>))	SI	SI
(CLEAR)	NO	SI
— SAY Muestra los datos formateados en la impresora	SI	SI
— GET Lee los datos formateados para la edición	SI	SI
ACCEPT (string) TO (var) Memoriza una cadena de caracteres en una variable de memoria	SI	SI
APPEND BLANK Añade registros vacíos al final del archivo	SI	SI
APPEND FROM <file> (FOR) <exp> (SDF) (DELIMITED) (WHILE) <exp> Añade registros en un archivo tomándolos de otro	SI	SI
ASSIST Ayuda a la ejecución de las órdenes del dBASE III	NO	SI
AVERAGE <lista exp> (<rango>) (FOR/WHILE <exp>) (TO <lista var>) Calcula las expresiones numéricas	NO	SI
BROWSE (FIELDS <lista fields>) Muestra una pantalla de 17 registros por cada archivo	SI	SI

SINTAXIS	dBASE II	dBASE III
CANCEL Suspende la ejecución del programa restableciendo el prompt	SI	SI
CHANGE (<rango>) FIELDS <lista> (FOR/WHILE<exp>) Escribe los datos en campos específicos	SI	SI
CLEAR Borra la pantalla	NO	SI
CLEAR ALL Cierra todos los archivos DBF, index, formal y relacionales. Además limpia todas las variables y selecciona el área de trabajo	NO	SI
CLEAR GETS Anula todas las variables GET	SI	SI
CLEAR MEMORY Borra todas las variables de memoria	SI	SI
CLOSE FORMAT/INDEX/PROCEDURE) Cierra los archivos especificados	NO	SI
CONTINUE Continúa la búsqueda en los archivos después de la orden LOCATE	SI	SI
COPY FILE <file name> TO <file name> Duplica cualquier tipo de archivo	NO	SI
COPY TO <file name> (<rango>) (FIELDS <lista fields> (FOR/WHILE <exp> (SDF/ DELIMITED) (WITH <delimited>))) Copia el archivo en USE en otro archivo DBF o text, cuando se utilizan las opciones SDF o DELIMITED	SI	SI
COPY TO <file> STRUCTURE (EXTENDED) (FIELDS <lista fields>) Duplica la estructura del archivo en USE en un nuevo archivo	SI	SI
COUNT (<rango>) FOR/WHILE <exp>) (TO <var>) Cuenta el número de registros especificados en el "rango"	SI	SI
CREATE (.dbf file name) (<file name> FROM <file name>) Define la estructura de un archivo y lo añade al directorio	SI	SI
CREATE LABEL <.lbl file name> Crea un archivo tipo label	NO	SI

SINTAXIS	dBASE II	dBASE III
CREATE REPORT Crea un archivo de tipo REPORT	NO	SI
DELETE (<rango>) (FOR/WHILE <exp>) Marca los registros especificados para el borrado	SI	SI
DELETE FILE <filename> Borra el archivo especificado	SI	SI
DIR (<drive:) <letras> <máscara> Lista los nombres de los archivos que se encuentran en el drive especificado	NO	SI
DISPLAY (<máscara>) (FIELD) <lista fields> (FOR/WHILE) <exp> (OFF) Lista los registros del archivo activo abierto	SI	SI
DISPLAY FILES (ON <disk drive>) (LIKE <máscara> Lista todos los archivos presentes en el disco o bien sólo aquellos con la extensión indicada	NO	SI
DISPLAY MEMORY Lista las variables de memoria	SI	SI
DISPLAY STATUS Lista el estado de un archivo y todos sus derivados, index, alternate y parámetros de sistema	NO	SI
DISPLAY STRUCTURE Visualiza la estructura del archivo	SI	SI
DO <prg file name>/<procedure> (WITH <lista parámetros>) Ejecuta un programa o un procedimiento y, opcionalmente, pasa los parámetros al programa	SI	SI
DO CASE..CASE (OTHERWISE)... ENDCASE Ejecuta sólo una de las posibles acciones instalada en el procedimiento. Se puede introducir un procedimiento alternativo con OTHERWISE. Se concluye todo con ENDCASE	SI	SI
DO WHILE <comandos>..ENDDO Permite la estructuración de procedimientos; salta todas las órdenes entre la condición no cumplida y ENDDO. DO WHILE tiene que acabar con ENDDO	SI	SI
EDIT (RECORD) <n> Permite cambios en el contenido de los campos	SI	SI

SINTAXIS	dBASE II	dBASE III
EJECT Hace avanzar el papel una hoja	SI	SI
ERASE <file name> Borra los archivos especificados	NO	SI
ERASE Borra la pantalla	SI	NO
EXIT Acaba un LOOP DO sin terminar la ejecución del programa	NO	SI
FIND <string> Coloca el cursor en la primera cadena con los caracteres especificados que encuentra	SI	SI
GO/GOTO BOTTOM/BOTTOM/TOP/ <exp n> Coloca el puntero en el registro especificado	SI	SI
HELP (<command>) Explica las órdenes dadas del dBASE III	NO	SI
IF...(ELSE)...ENDIF Permite la ejecución condicional de órdenes en un programa y, opcionalmente, con la alternativa ELSE. Cada IF tiene que terminar con ENDIF	SI	SI
INDEX ON <string,exp> TO <index file name> Cataloga un archivo según una clave especificada	SI	SI
INPUT (<string>) TO <var> Permite insertar una cadena de caracteres en la variable de memoria	SI	SI
INSERT (BLANK) (BEFORE) Inserta un registro en una posición específica del archivo	SI	SI
JOIN TO <nuevo archivo> FOR <condición> (FIELD <lista fields>) Une dos archivos combinando los campos en un tercero	SI	SI
LABEL FORM <lbl file name> (SAMPLE) (<rango>) (FOR/WHILE <condición>) (TO PRINT/TO FILE <file name>) Imprime las etiquetas usadas en los archivos especificados	NO	SI
LIST (FOR/WHILE condition) (FIELDS <lista fields>) (OFF) Lista los campos de un archivo	SI	SI

SINTAXIS	dBASE II	dBASE III
LIST MEMORY/STATUS/STRUCTURE Igual que el DISPLAY MEMORY, STATUS o STRUCTURE	SI	SI
LOCATE (<rango>) FOR <condición> Coloca el puntero en el registro que satisface la condición especificada	SI	SI
LOOP Salta todas las órdenes situadas entre LOOP y ENDDO	SI	SI
MODIFY COMMAND <file name> Permite la edición de archivos con caracteres ASCII, incluyendo los programas en los archivos (.prg)	SI	SI
MODIFY LABEL <.lbl file name> Permite modificar un archivo de tipo LBL con la plantilla en la pantalla	NO	SI
MODIFY REPORT <.frm file name> Permite modificar un archivo de tipo .FRM con la plantilla en la pantalla	NO	SI
MODIFY STRUCTURE <file name> Modifica la estructura de un archivo .DBF en uso	SI	SI
NOTE/* <caracteres y cadenas ilimitadas> Inserta comentarios en el programa	SI	SI
PACK Elimina los registros marcados del disco	SI	SI
PARAMETERS <lista parámetros> Especifica las variables de memoria que usan informaciones pasadas por la orden DO...WITH	NO	SI
PRIVATE (ALL(LIKE/EXCEPT)) Transfiere las variables de memoria a otro nivel de memoria	NO	SI
PUBLIC <memory variable list> Hace aparecer todas las variables de memoria	NO	SI
QUIT Cierra todos los archivos y sale del sistema dBASE	SI	SI
READ Permite la lectura de las variables insertadas en el campo mediante GET	SI	SI
RECALL (<rango>) (FOR/WHILE <condición>) Reinstala los registros marcados para el borrado	SI	SI
REINDEX Recataloga un archivo INDEX	SI	SI

SINTAXIS	dBASE II	dBASE III
RELEASE <lista var> (ALL) (ALL like/except) (ALL EXCEPT <condición>) Borra las variables de memoria	SI	SI
REMARK Aumenta la intensidad en la pantalla	SI	NO
RENAME <file name actual> TO <nuevo file name> Da al archivo un nuevo nombre	SI	SI
REPLACE (<rango>) (field) WITH <exp> (,<field WITH (exp2>,...) (FOR/WHILE <condición>) Modifica el contenido de los campos especificados	SI	SI
REPORT FORM <.form file name> (<rango>) (FOR <exp>) (PLAIN) (HEADING <string> NOEJECT) (TO PRINT) (TO FILE <file name>) Formatea un tabulado de datos	SI	SI
RESET (<drive>) Reinstala el programa	SI	SI
RESTORE FROM <file menu name> (ADDITIVE) Llama las variables de la memoria	SI	SI
RETURN (TO MASTER) Concluye el programa. La línea donde está RETURN es la última línea ejecutable	SI	SI
RUN <command> Ejecuta un programa fuera del dBASE	NO	SI
SAVE TO <.mem file name> (ALL LIKE/ EXCEPT <máscara>) Copia las variables de memoria en un archivo	SI	SI
SEEK <expresión> Coloca el puntero en el primer registro con una clave índice que corresponda a la expresión especificada	NO	SI
SELECT (n/alias) (PRIMARY) (SECONDARY) Abre archivos simultáneamente (en el dBASE II n. hasta 2; en el dBASE III n. hasta 10	SI	SI
SET Instala los parámetros del dBASE TO para indicar una acción a ejecutar ON para activar una operación OFF para desactivar una operación	SI	SI

SINTAXIS	dBASE II	dBASE III
SET ALTERNATE TO <file name> Crea un archivo para guardar una salida	SI	SI
SET ALTERNATE on/OFF Transmite/NO TRANSMITE el archivo de salida	SI	SI
SET BELL ON/off Conecta/DESCONECTA el sonido durante la entrada de datos	SI	SI
SET CARRY on/OFF Escribe/NO ESCRIBE el contenido del último registro añadido con APPEND	SI	SI
SET COLOR TO <standard display> <change of display> (<border>) Instala la visualización en color	NO	SI
SET CONFIRM on/OFF Desactiva/ACTIVA el salto automático después de introducir una variable sin RETURN	SI	SI
SET CONSOLE ON/off TRANSMITE/suspende todas las uniones entre el teclado y la pantalla	SI	SI
SET DATE DO Instala el formato de la fecha	SI	NO
SET DEBUG on/OFF Transmite/SUSPENDE la salida del DEBUG a la impresora	SI	SI
SET DECIMALS TO n Instala el número mínimo de decimales en el resultado de determinadas operaciones y funciones	NO	SI
SET DEFAULT TO Especifica el drive en funcionamiento	SI	SI
SET DELETED on/OFF Suspende/PROCEDE a la demarcación de los registros para el borrado	NO	SI
SET DELIMITER TO <string> (DEFAULT) Especifica la delimitación para la visualización de los campos y las variables	NO	SI
SET DELIMITER on/OFF Visualiza campos y variables delimitados en modo normal/EN MODO REVERSE	SI	SI
SET DEVICE TO SCREEN/print Transmite las coordenadas de la orden nnSAY' a la PANTALLA/a la impresora	NO	SI
SET ECHO on/OFF Hace aparecer/NO HACE aparecer las órdenes de la pantalla en la impresora	SI	SI

SINTAXIS	dBASE II	dBASE III
SET ESCAPE ON/off ANULA/continúa la ejecución del archivo command cuando se aprieta la tecla EXC	SI	SI
SET EJECT ON/off INSTALA/suspende el cambio de página automático	SI	NO
SET EXACT ON/off REQUIERE/no requiere la comparación exacta del carácter	SI	SI
SET FILTER TO Visualiza un archivo sólo si contiene los registros correspondientes a la condición	NO	SI
SET FIXED on/OFF Fija/NO FIJA el número de decimales a visualizar	NO	SI
SET FORMAT <.fmt file name> Abre un archivo FORMAT para la entrada de datos	SI	SI
SET FUNCTION <key #=> TO <string> Instala los valores de las funciones clave	NO	SI
SET HEADING on/OFF Centra/NO CENTRA las variables en la pantalla	SI	SI
SET HELP ON/off INSTALA/no instala el HELP cuando se cometen errores	NO	SI
SET INDEX TO <.ndx lista files> Abre los archivos catalogados	SI	SI
SET INTENSITY ON/off INSTALA/no instala el modo inverso en la pantalla	SI	SI
SET LINKAGE ON/OFF Conecta las órdenes secuencialmente ejecutándolas ya sea en el archivo primario, ya en el secundario. Mantiene independiente el archivo primario del secundario	SI	NO
SET MARGIN TO n Fija en "n" el margen izquierdo de la impresora	SI	SI
SET MENUS ON/off INSTALA/no instala la conexión entre un menú y otro	NO	SI
SET PATH TO <lista> Especifica el camino para la búsqueda del archivo	NO	SI

SINTAXIS	dBASE II	dBASE III
SET PRINT on/OFF Transmite/NO TRANSMITE la salida de datos a la impresora	SI	SI
SET PROCEDURE TO <procedure file name> Abre el archivo que contiene el procedimiento especificado	NO	SI
SET RAW on/OFF Instala/ANULA las columnas en las listas	SI	NO
SET RELATION TO <key> INTO <alias> Une dos archivos en base a la expresión de la clave	NO	SI
SET SAFETY ON/off INSTALA/no instala un nivel de protección contra la pérdida de datos	NO	SI
SET SCREEN ON/off INSTALA/suspende el uso de las órdenes EDIT, APPEND, INSERT a toda pantalla	SI	NO
SET STEP on/OFF Para/NO PARA la ejecución de un programa después de llevar a cabo cada orden	SI	SI
SET TALK ON/off TRANSMITE/no transmite a la pantalla los resultados de la ejecución de una orden	SI	SI
SET UNIQUE on/OFF Hace aparecer el primer registro con la misma clave de un archivo	NO	SI
SKIP n (+<exp>) (-<exp>) Coloca el puntero en el registro y lo hace avanzar y retroceder según las especificaciones dadas	SI	SI
SORT TO <new file name> ON <field> (/A) (/D) (,field2) (/A) (/D) (<rango>) (FOR <condición>) Crea un archivo ordenado, según un campo determinado	SI	SI
STORE <exp> TO <var> (<lista var>) Conserva una expresión en una o más memorias variables	NO	SI
SUM (<rango>) (<lista expresiones>) TO <lista var> (FOR/WHILE <condición>) Calcula y lista la suma de una expresión referente a los registros del archivo especificado en el rango	SI	SI
TEXT.ENDTEXT Visualiza una parte del texto de un archivo orden. El texto tiene que finalizar con ENDTEXT	NO	SI

SINTAXIS	dBASE II	dBASE III
TOTAL <file name> ON <key> <rango> (FIELD <lista fields>) (FOR/WHILE <condición>) Crea un sumario de un archivo pre-ordenado con SORT conteniendo totales numéricos	SI	SI
TYPR <filename> <exp> Sirve para mostrar el contenido de cualquier archivo en la pantalla o impresora	NO	SI
USE (<.dbf file name>) (INDEX <.ndx list files>) (ALIAS <alias>) Especifica el archivo a utilizar para todas las operaciones hasta que no sea admitido otro USE	SI	SI
WAIT (<promp>) (TO <var>) Suspende el proceso del programa hasta que no se apriete la clave	SI	SI
ZAP Borra todos los registros del archivo abierto	NO	SI

De entre todos los programas de bases de datos realmente dignos de tal nombre, el dBASE es, sin duda, el más conocido.
En este libro veremos cómo tanto el dBASE II como el dBASE III nos pueden ayudar, con su potencia y flexibilidad, a solventar todos aquellos problemas en los cuales la intervención de una base de datos se vuelve algo imprescindible.
Crear y estructurar archivos, manipular y actualizar los datos en ellos incluidos, sacar informes y resúmenes de los mismos, establecer relaciones entre ellos, aprovechar el sistema gestor de la base de datos..., todos estos temas y muchos más serán tratados en este volumen de la Biblioteca Básica Informática.

395 pts.
(incluido IVA)